Hierarchical control of large cryogenic refrigerators

Xuan-Huy Pham

To cite this version:

Xuan-Huy Pham. Hierarchical control of large cryogenic refrigerators. Automatic. Université Grenoble Alpes [2020-..], 2022. English. NNT : 2022GRALT105 . tel-04051565

HAL Id: tel-04051565

https://theses.hal.science/tel-04051565v1

Submitted on 30 Mar 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

École doctorale : EEATS - Electronique, Electrotechnique, Automatique, Traitement du Signal (EEATS)
Spécialité : Automatique - Productique
Unité de recherche : CEA Grenoble (hors LETI et LITEN)

Contrôle hiérarchisé des grands réfrigérateurs cryogéniques

Hierarchical control of large cryogenic refrigerators

Présentée par :

Xuan-Huy PHAM

Direction de thèse :

Mazen ALAMIR Directeur de thèse
DIRECTEUR DE RECHERCHE, Université Grenoble Alpes

Rapporteurs :

Romain BOURDAIS
PROFESSEUR DES UNIVERSITES, CENTRALESUPELEC
Gabriele PANNOCCHIA
PROFESSEUR, Università degli Studi di Pisa

Thèse soutenue publiquement le **15 décembre 2022**, devant le jury composé de :

Mazen ALAMIR Directeur de thèse
DIRECTEUR DE RECHERCHE, CNRS DELEGATION ALPES
Romain BOURDAIS Rapporteur
PROFESSEUR DES UNIVERSITES, CENTRALESUPELEC
Gabriele PANNOCCHIA Rapporteur
PROFESSEUR, Università degli Studi di Pisa
Benjamin BRADU Examinateur
INGENIEUR DOCTEUR, CERN
Gildas BESANCON Président
PROFESSEUR DES UNIVERSITES, GRENOBLE INP

Invités :

François BONNE
INGENIEUR DOCTEUR, CEA-Grenoble

Remerciements

First of all, I would like to thank everyone who has accompanied me during my life and research at Grenoble.

Most especially, I would like to express my sincere thanks to the people who guided me to complete my PhD thesis, Professor Mazen Alamir, Mr. Patrick Bonnay and Mr. François Bonne. Thanks to their enthusiastic guidance and rich experience, I was able to complete this thesis, especially during the writing phase, I am very touched by their patience with me.

I would like to thank all my colleagues at DSBT who were very nice people who accompanied me during my research: Jeyathasan, Christiophe, Anthony, Julien, Thomas, ... Thanks to them, I am always motivated to go to work every morning and their stories helped me adjust to life in France.

I would like to thank those who examined my thesis and agreed to participate in my dissertation review committee, professor Romain Bourdais, professor Gabriele Pannocchia, professor Gildas Besancon and Dr. Benjamin Bradu. Their comments have helped to improve the quality of my thesis.

I want to thank my family, my grandfather, my parents and two sisters for always supporting my choices and always accompanying me. I wish them good health and always be with me every step of the way.

Finally, I would like to thank my dear girlfriend Uyen, who has been with me and cheered me on and always knows how to make me laugh and pull me away from work to rest. Without you, my thesis would have been much more difficult.

Contents

4 On the use of fast-NMPC and deep learning approach in fixed-point-based hierarchical control 73

List of Figures

List of Tables

Table of abbreviations and acronyms

Acronyms

BCE	Before the Christian Era
CEA	Commissariat à l’Énergie Atomique et aux l’Énergie Alternatives
CERN	European Council for Nuclear Research
DNN	Deep neural network
DSBT	Département des systèmes à basse température
ITER	International Thermonuclear Experimental Reactor
GLM	Generalized linear models
JT	Joule-Thomson
SL	Saturated liquid
SV	Saturated Vapor
MPC	Model predictive control
MSE	Mean squared error
NMPC	Nonlinear model predictive control
NLR	Nonlinear regression
PID	Proportional-Integral-derivative
PLC	Programmable Logic Controller
RF	Random forest
SVM	Support vector machine
VAR	Vector autoregressive model
WCS	Warm Compression Station

Physical quantity

C_p	Specific heat
P	Pressure
M	Flow rate
T	Temperature
μ	Efficiency index
γ	specific heat ratio
Q	specific heat ratio
h	specific enthalpy
s	specific entropy
$CV_{\max}$	Flow rate coefficient of a valve
R_V	Rangeability of a valve
ρ	Mass density

Indexes and Exponents

L	Refers to the property of a fluid under low pressure
H	Refers to the property of a fluid under high pressure

Résumé en Français

Cette section résume brièvement le contenu principal de ce livre, chapitre par chapitre, qui sont énumérés ci-dessous:

Chapter 2: *Background*

Nous présentons les préliminaires essentiels à notre recherche principale, qui est l'approche de contrôle hiérarchique dans les processus cryogéniques. Tout d'abord, nous introduisons la méthode de décomposition qui est nécessaire pour séparer le système global en de nombreux sous-systèmes interagissant entre eux via des signaux de couplage. Puisque le sujet de ce livre est de proposer une méthode de contrôle pour les systèmes à grande échelle, le cadre général de contrôle décentralisé basé sur des observateurs est décrit, ce qui sert de référence pour être comparé à la méthode proposée tout au long de ce livre. Les contrôleurs locaux des sous-systèmes de ce cadre peuvent être choisis comme la commande prédictive, bien connue et largement utilisée, dont la formulation est ensuite rappelée. Enfin, le schéma d'observation centralisé pour l'estimation d'état est décrit.

Chapter 3: *Fixed-point-iteration-based hierarchical control*

Nous présentons le cadre de contrôle hiérarchique basé sur le point fixe, qui consiste en deux couches distinctes (Fig. 1). À la couche inférieure, les sous-systèmes interconnectés sont contrôlés par des agents locaux, qui ont besoin de profils de couplage pour calculer leurs profils de contrôle. À la couche supérieure, le coordinateur communique abondamment avec les agents pour trouver les véritables profils de couplage associés à un point de consigne donné. En outre, le coordinateur optimise un coût central par rapport aux points de consigne qui seront envoyés aux agents. Ce problème d'opimisation est donné par l'équation ci-dessous:

$$r^{\text{opt}} = \underset{r \in \mathcal{R}}{\text{argmin}} \ J_c(r, \boldsymbol{v}^{in}) \tag{1}$$

$$\text{subject to: } \boldsymbol{v}^{in} = \boldsymbol{g}^{in}(r, \boldsymbol{v}^{in}) \tag{2}$$

où $\mathcal{R}$ est l'ensemble admissible de r et $\boldsymbol{v}^{in}$ est le profil de couplage global des sous-systèmes. Afin de re-formuler ce problème, le lecteur est renvoyé au chapitre 3. Il faut tenir compte du fait que le coordinateur ne connaît pas les informations des sous-systèmes (ses modèles ou ses contrôleurs). A partir des équations (1)- (2), nous pouvons voir que cela prend la forme

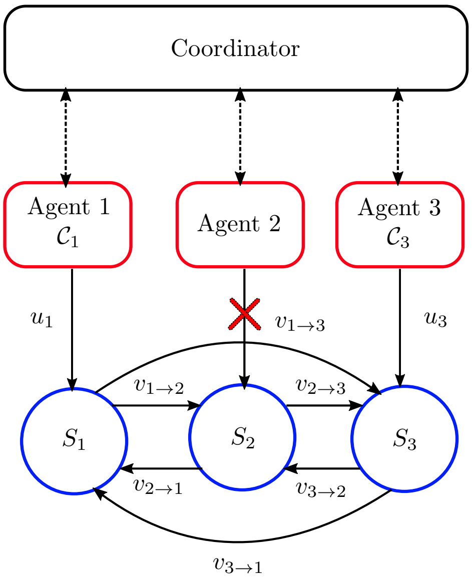


Figure 1: Structure hiérarchique appliquée à un réseau de trois sous-systèmes interconnectés. Les lignes de tirets à double flèche représentent la communication entre le coordinateur et les agents.

d'un problème du point fixe. Pour résoudre ce problème, un algorithme est proposé dans le chapitre 3.

Dans le travail original [1], ce cadre a été validé dans le cas où les agents locaux implémentent uniquement des contrôleurs linéaires non contraints. Nous avons étendu ce travail en remplaçant ces contrôleurs par des contrôleurs non linéaires contraints et en évaluant l'efficacité du framework dans ce cas. En outre, le problème d'implémentation en temps réel résultant de l'utilisation de contrôleurs non linéaires est également abordé en proposant une technique qui consiste à répartir l'optimisation dans le temps. En additon, dans le travail original, ce cadre de contrôle a été appliqué pour contrôler la boîte froide qui est décomposé en deux sous-système. Dans le travail présent, cette méthode est appliquée pour une décomposition avec plus de deux sous-systèmes (quatre sous-systèmes pour être précis) afin de évaluer la performance en termes de contrôle et le temps de calcul. Fig; 2 montre la boîte froide (à gauche) et ses décompositions (à droite) intéressées par ce chapitre. Enfin, une série de simulations montre l'efficacité du cadre proposé.

Chapter 4: *On the use of fast-NMPC and deep learning approach in fixed-point-iteration-based hierarchical control*

Dans ce cadre, les problèmes MPC non linéaires contraints sont résolus de manière répétée jusqu'à ce que la communication entre le coordinateur et l'agent local converge vers un point fixe, ce qui augmente définitivement la charge de calcul. En effet, il y aura un goulot

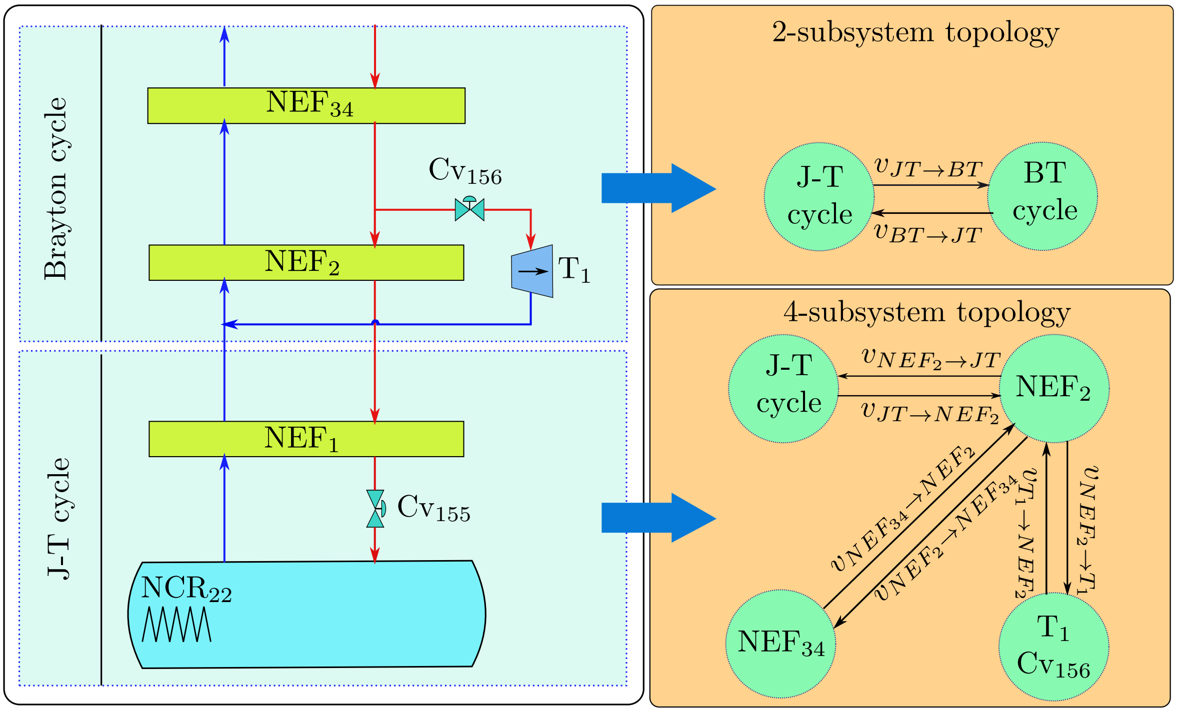


Figure 2: Two possible decompositions of the coldbox of the 400 W station.

d'étranglement pour la mise en œuvre en temps réel si le temps de calcul requis est éloigné de la période de mise à jour. Afin de réduire le temps de calcul des contrôleurs locaux, des techniques MPC non linéaires rapides telles que le solveur de gradient tronqué et le contrôleur basé sur un réseau neuronal profond sont proposées. Tout d'abord, le solveur basé sur descente de gradient est implémenté pour résoudre les problèmes d'optimisation des commandes prédictives synthétisées pour le cycle Joule-Thomson et la turbine. Table 1

Table 1: Indice de performance du solveur Ipopt et du solveur basé sur le gradient tronqué.

	NMPC of S_1				NMPC of S_4			
Solver	$N_{\max}$	ϵ_{tol}	$\bar{J}\,[\%]$	$t_{\text{cpt}}^{\max}$ [s]	$N_{\max}$	ϵ_{tol}	$\bar{J}\,[\%]$	$t_{\text{cpt}}^{\max}$ [s]
Truncated gradient descend	100	_	100.2379	0.0499	100	_	101.3418	0.011
	50	_	100.2395	0.0398	50	_	101.3419	0.008
	30	_	100.2357	0.0322	30	_	101.3418	0.0043
	10	_	101.29	0.0246	10	_	101.3418	0.0014
Ipopt/Casadi	5	10^{-1}	99.999	2.746	5	10^{-1}	100	0.0589
	10	10^{-1}	100.002	3.756	10	10^{-1}	100	0.0873
	10	10^{-4}	100	4.76	10	10^{-4}	100	0.1720

Ensuite, après l'intégration de la commande prédictive dans le cadre de la commande hiérarchique grâce à l'utilisation du solveur basé sur le gradient, elles sont approximées par des réseaux de neurones afin de réduire davantage le temps de calcul et de rendre cette méthode implémentable dans des automates industriels. Nous proposons la procédure de génération de données suivante qui effectue une simulation hors ligne à l'aide du modèle de système sous la loi de commande afin de collecter l'ensemble de formation $\mathcal{D}$ pertinent sur le plan opérationnel :

1. Déterminer la plage opérationnelle des points de consigne désignée par $[\underline{r}, \overline{r}]$ et la plage réaliste des perturbations désignée par $[\underline{w}, \overline{w}]$:
2. Créer des signaux binaires pseudo-aléatoires (PRBS) de r dans ses plages opérationnelles. Afin de capturer le comportement de suivi de la consigne du contrôleur, l'amplitude du signal ne doit pas varier pendant une période de temps suffisante désignée par Δt. La forme du signal de perturbation w peut être choisie en fonction de son comportement réaliste. Dans notre application, la forme du signal w et la période de temps Δt seront spécifiées dans la Seconde partie. 4.4.3.1.
3. Exécutez les simulations en boucle fermée qui mettent en œuvre la conception hiérarchique susmentionnée à certains états initiaux choisis avec les signaux PRBS créés. Notez que les données sont collectées pendant les itérations à point fixe afin de capturer la relation entre le profil de contrôle $\boldsymbol{u}$ et le triplet $(r, x, \boldsymbol{v}^{in})$.

Le réseau est entraîné pour minimiser le critère d'erreur quadratique moyenne ci-dessous:

$$J_{NN}(\theta) = \frac{1}{2}\sum_{i=1}^{N_{tr}} \|\boldsymbol{u}^{(i)} - K_{NN}(\xi^{(i)}, \theta)\|^2 \tag{3}$$

où $N_{tr} < N$ est le nombre d'observations d'apprentissage et N est le nombre d'observation créées par le processus. En effet, avant le processus d'apprentissage, l'ensemble de données passe par une série de techniques de préparation des données et est finalement séparé en deux sous-ensembles qui contiennent N_{tr} échantillons et $N_{val} = N - N_{tr}$ échantillons, qui servent à entraîner et à valider le modèle de régression. Rappelons que le vecteur $\xi^{(j)}$ encapsule tous les paramètres $x^{(i)}$, $\boldsymbol{v}^{in,(i)}$, $r^{(i)}$ et $\boldsymbol{w}^{(i)}$.

Il est également démontré qu'en procédant ainsi, la période de mise à jour du contrôle peut être réduite de manière significative et les performances en boucle fermée grandement améliorées. Ce chapitre peut donc être considéré comme une mise en œuvre concrète et une validation de certaines idées clés dans la conception de NMPC distribués en temps réel.

Chapter 5: *Application to a complete refrigerator and some development in the foxed-point based hierarchical control method*

Le cadre proposé est appliqué au système complet, qui se compose de huit sous-systèmes avec des échelles de temps différentes. Figure. 3 montre la decomposition réalisée pour le réfrigérateur cryogénique de 1000 W à 4,4 K.

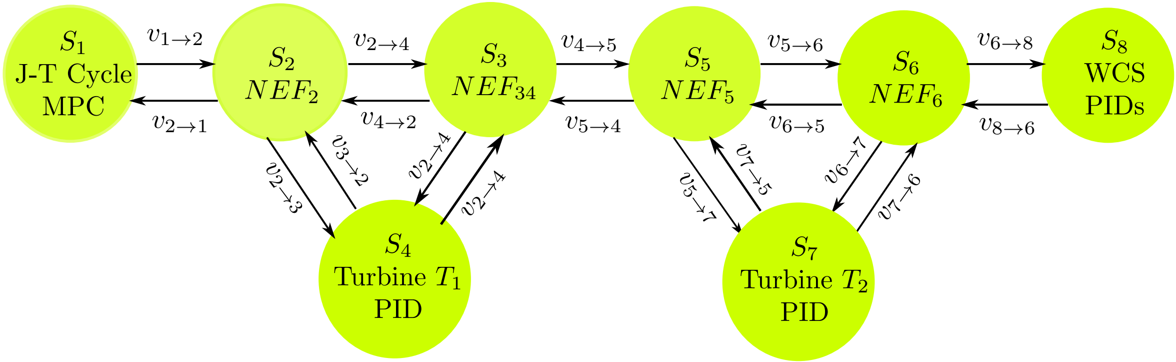


Figure 3: L'interconnexion entre les sous-systèmes de l'installation cryogénique.

Nous pouvons voir que ce système est décomposé en huit sous-systèmes avec les différents contrôleurs. Le cycle de Joule-Thomson est contrôlé par le contrôleur prédictif et les deux turbines T_1 et T_2 sont contrôlées par le contrôleur PID. Ces trois sous-systèmes fonctionnent avec le temps de mise à jour $\tau_u = 5s$. La station de compression (WCS), quant à elle, est contrôlée par la méthode du split-range et fonctionne avec un temps de mise à jour plus rapide $\tau_u = 1s$. Pour implémenter ce cadre de contrôle hiérarchisé, nous devons proposer une hypothèse sur le fonctionnement des sous-systèmes qui sera présentée dans le chapitre 5.

Dans le travail précédent, la convergence de l'itération du point fixe est amélioré par un filtre qui est synthétisé à partir des informations privées des sous-système. Cela viole l'exigence de préservation de la confidentialité modulaire. Ainsi, une méthode itérative basée sur les résidus, appelée méthode d'accélération d'Anderson, est mise en œuvre pour assurer la convergence de la boucle interne traitée entre le coordinateur et les agents. Cette méthode utilise uniquement l'information historique sur les résidus pendant l'itération pour améliorer sa convergence. Dans la section de simulation, il est montré que la méthode avec le filtre ne marche pas pour ce problème de contrôle, alors que la méthode d'Anderson améliore la convergence de l'intération du point-fixe.

En outre, le problème devient complexe cars il y a plus de consignes à optimiser par le coordinateur. Un solveur basé sur le gradient est mis en œuvre au niveau de la coordination pour optimiser le coût central par rapport au point de consigne. Des simulations numériques sont effectuées pour évaluer l'efficacité du cadre proposé.

Publication related to this manuscript

International journals

X. H. Pham, M. Alamir, F. Bonne, and P. Bonnay. "Revisiting a fixed-point hierarchical control design for cryogenic refrigerators under constraints, nonlinearities and real-time considerations". *European Journal of Control*, 2022, vol. 63, p. 82-96.

X. H. Pham, F. Bonne and M. Alamir. "On the use of gradient-based solver and deep learning approach in hierarchical control: Application to grand refrigerators", 2022.(submitted to *Cybernetics and Systems*).

International conferences

X. H. Pham, M. Alamir and F. Bonne, "Investigation of fast-NMPC and deep learning approach in fixed-point-based hierarchical control," *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2022, pp. 605-610.

X. H. Pham, M. Alamir, F. Bonne and P. Bonnay, "Using iterative residual-based method for modular privacy-preserving requirement in hierarchical control framework", *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2022, pp. 718-723, doi: 10.1109/CoDIT55151.2022.9804044.

X. H. Pham, M. Alamir, F. Bonne and P. Bonnay, "Computation time reducing via non linearity isolating and constraint handling in fixed-point based hierarchical control framework", *2022 IEEE 17th International Conference on Control & Automation (ICCA)*, 2022, pp. 921-926, doi: 10.1109/ICCA54724.2022.9831869.

CHAPTER 1

General Introduction

Abstract This chapter begins by presenting the energy context, where the importance of fusion energy and the need for cryogenic applications in fusion energy production are described. Later, the helium refrigerator and its subsystems are introduced. Several control strategies for such cryogenic processes are carefully listed and described. Finally, the chapter concludes by evoking the problem of these conventional control strategies.

1.1 Introduction

1.1.1 Motivation

About 1.5 million years ago, a spark ignited the first human-made fire and sparked a great revolution for our species: an energy revolution. Our ancestor used wood to fuel fire for many uses such as cooking and additional source of heat and light. In 300 BCE, humans learned how to harness the energy of flowing water. They used water wheels to draw water from river in order to fill aqueducts, to irrigate farmland and to mill flour. In the 1800's, humankind saw it second energy revolution by discovering the use of fossil fuels. This discovery spurred industrial revolution that transformed the world with new mechanical and electrical technologies emerged from steam engines to electric turbines. At the same time, natural gas was discovered and used in Britain and United States for addition lighting in the evenings. This extended the human productivity deep into the night. After the discovery of electricity, the human civilization has developed rapidly and obtained outstanding achievements over the centuries. Clearly, the discovery of energy is one of the most radical transformations in human history.

Nowadays, the development of society leads to a significant demand for energy for activities and production. Illustratively, Fig 1.1 shows the global energy production that increases continuously from 1985 to 2020. However, more than a half of the energy produced comes from fossil fuels, such as coal, natural gas, and oil, which are essentially limited resources and considered as high emitters of greenhouse gases.

In the context of increasingly severe climate change, although the promotion of the production and use of renewable energy increases strongly, the rate remains limited for

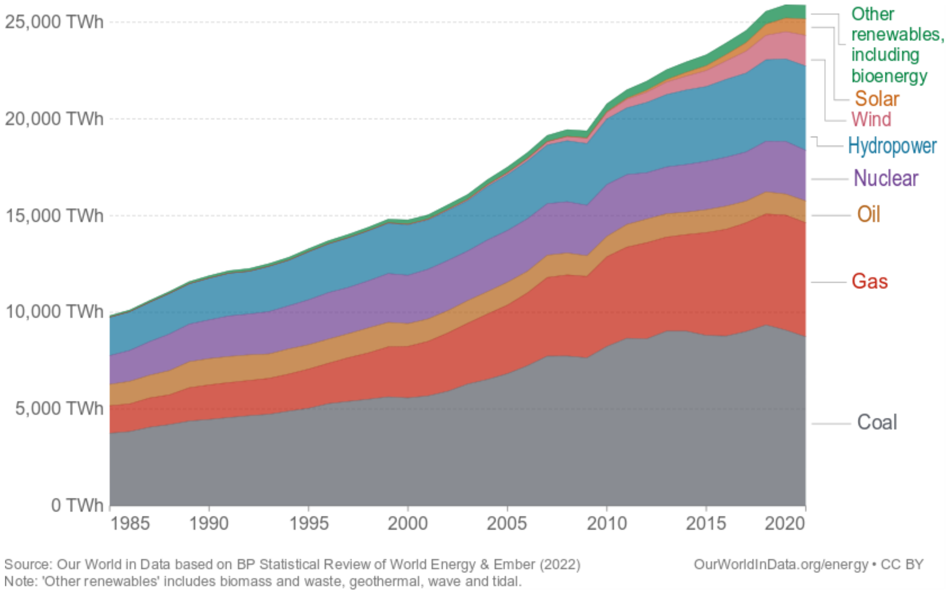


Figure 1.1: World energy production by source from 1985 to 2020 [2]

apparent reasons such as territory footprint, pollution, storage problem, etc. Indeed, a 1000 Megawatt (MW) solar facility requires between 117 km^2 and 194 km^2 while a equivalent nuclear energy facility requires 3.37 km^2 [3]. In addition, the production of some photovoltaic cells that are used in solar panels generates toxic substances that may contaminate water resources. Furthermore, most of renewable energy is intermittent and it requires large batteries in order to store any additional produced electricity. These reasons are barriers preventing the spread of renewable energy.

Nuclear energy could be considered as a "clean" option as it produces fewer greenhouse gas compared to other sources (Fig 1.2). Despite producing massive amount of carbon-free power, nuclear energy produces more electricity on less land than any other clean-air source. Unfortunately, after the catastrophes in Chernobyl (1986) and Fukushima (2011), the world has changed its view on this energy because of safety reasons. Instead, a new type of energy produced by the nuclear fusion reaction becomes an alternate option for future energy supply, which possibly addresses all the discussed drawbacks of the existing energy sources.

In the next section, we will discuss the technology behind fusion reactors and how vital that the use of cryogenic refrigerators is in the nuclear fusion context.

1.1.2 Principles of fusion reactors

Despite the fact that fusion is the most dominant reaction in our universe, the science of nuclear fusion only got more attention when British astrophysicist Arthur Eddington suggested in 1920 that stars draw their energy from the fusion of *hydrogen* into helium [5].

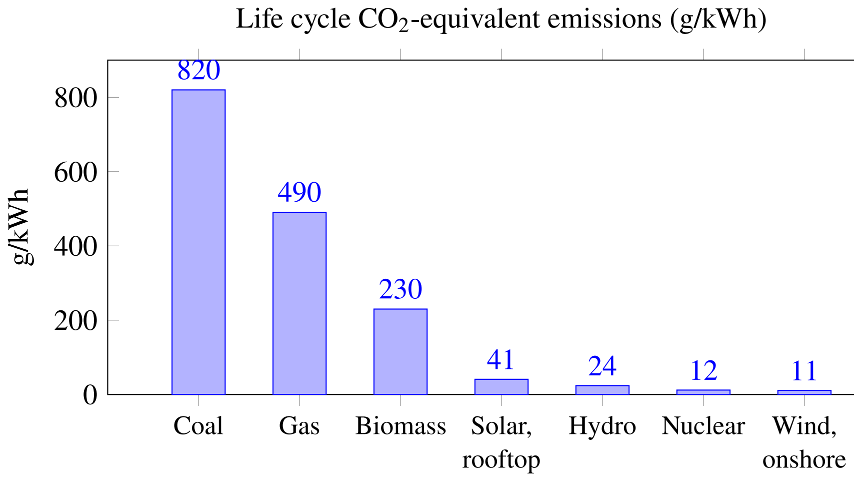


Figure 1.2: Life cycle greenhouse gas emission of selected electricity supply technologies [4].

Fusion reaction is a thermonuclear process where nuclei merge and fuse together, creating heavier nuclei and releasing energy in the process. It is also the energy that the scientists want to harness. In order for this reaction to occur, a certain amount of kinetic energy must be supplied to cause the particles to come together. It has been shown that the most probable reaction is the one between deuterium and tritium [6]. Deuterium can be found in seawater with an estimated concentration of 33 g/m^3, while the tritium can be produced from lithium presenting in hard rock and seawater (0.17 g/m^3). With these isotopes (deuterium (D) and tritium (T)), the following reactions may occur [7]:

$$D + T \rightarrow {}^4He + n(14MeV)$$
$$D + D \rightarrow {}^3He + n(2.45MeV)$$
$$D + D \rightarrow T + p(3.02MeV)$$

Since these nuclei are positively charged and therefore repel each other, high energy is required to overcome this repulsive force. In this context, this energy exists in the form of heat of millions of degrees. When a gas is subjected to such extreme temperature, the electrons are separated from the nuclei, and the gas turns into plasma, the fourth state of matter. Effectively, plasma provides the environment where nuclei can fuse and generate energy.

Three conditions must be met to obtain fusion: a very high temperature (to cause high-energy collisions), a sufficient density of plasma particles (to increase the probability of collisions), and a sufficient confinement time (to maintain the plasma, which tends to expand).

Until now, there are two major research branches that study how to make plasmas hot enough to fuse, namely:

- Magnetic confinement technology: In this technology, a doughnut-shaped chamber based on superconducting materials is constructed. Hydrogen fuel that contains deuterium and tritium is injected inside the chamber and is placed under extreme heat and low pressure that turn it into plasma. The electrons produced by the fusion reaction in the plasma can collide into the chamber and heat up the material, which can cause the facility to stop. Since plasma is a mix of positively charged ions and negatively charged electrons, a powerful electric current is passed through the superconducting materials to create a magnetic field having the same shape of the container, which can confine the plasma inside the chamber.

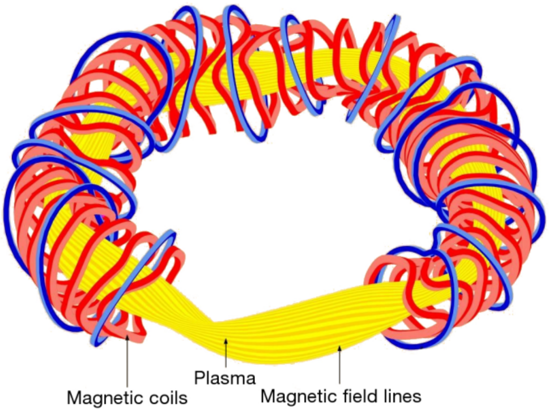
- Inertial confinement technology: This technology uses pulses from a high-power laser to compress targets filled with fuel. The targets are small, pinhead-sized spherical pellets typically containing a mixture of deuterium and tritium. The laser beams heat the outer layer, which consequently explodes outward, producing a reaction force against the remainder of the target. This reaction accelerates the outer layer inward and compresses the fuel, making it explode and become hot and dense enough to fuse.

In the first technology, the plasma confinement chamber is based on the components made from superconducting materials. These components require cryogenic refrigerators to cool down their temperature to maintain their superconducting properties. The next section will describe magnetic confinement reactors and why cryogenic refrigerators are essential.

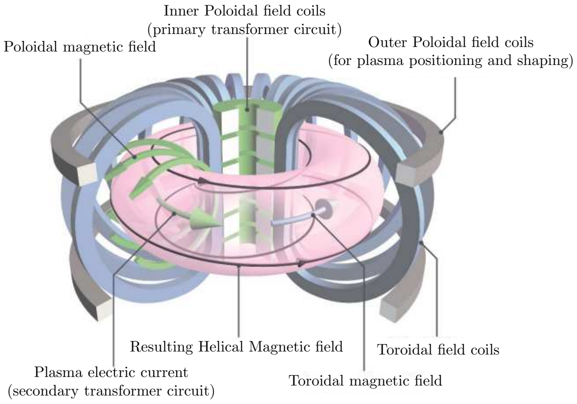
1.1.3 Magnetic confinement reactors

As previously described, this type of fusion reactor uses a magnetic field technology to heat and confine plasma in its chamber. More specifically, the plasma contains charged particles that are directed by the magnetic field. In others words, the plasma can be confined by creating a magnetic field in a particular shape. Fig. 1.3 shows field geometries that was proposed: Tokamak and stellarator. In Tokamak, the plasma current generating a poloidal magnetic field that is used, in combination with a toroidal field, to achieve the magnetic confinement configuration. While in stellarators, coils are designed in complex geometries to obtain magnetic confinement without plasma current.

In JET- the biggest operating tokamak in the world, copper electromagnets are used in order to obtain magnetic fields, which require a power up to 150 MW. A portion of this power is unintentionally decayed by the electrical resistivity in the material (Joule effect). This power is considered as a consumption waste. Moreover, the heat produced during the operation is not bearable, which can melt the cooper-based magnets. In order for the economics of fusion be viable and for preventing material melting subjected to extreme heat, superconductor-based magnets, which provide beneficial characteristics, are considered and used in many recently constructed tokamaks. Unlike ordinary metallic magnets, the electrical resistance of a superconductor drops abruptly to zero when its temperature is below the characteristic critical one. In addition, the superconducting magnets are constantly maintained at low temperature,



(a) Stellarator [8]



(b) Tokamak [9]

Figure 1.3: Magnetic confinement

the melting phenomena is prevented. All the large fusion reactors built since 1980s, such as the LHD stellarator (Japan), the tokamak Tore Supra (France), EAST (China), are equipped with superconducting magnets. Furthermore, under construction projects such as JT-60SA (Japan) and ITER will also rely on superconducting material to carry higher current and produce stronger magnetic field. It should be said that if the ITER's magnet were made from copper, a power of 800 MW is needed to supply them, whereas with superconducting technology, a smaller amount of power (20 MW) is required for the cryogenic plant to cool down the magnets.

In order for the superconductivity to occur, some conditions needs to be fulfilled, which is: the material temperature (T), the magnetic field (B) applied to the magnet and the magnitude of the current density (J) must be below some critical values. For instance, Fig. 1.4 illustrates the condition region, in which the niobium-titanium-alloy-based magnet, which is used for the tokamak Tore Supra, the tokamak KSTAR and the tokamak ITER, becomes super conductor. It can be noted that this alloy becomes superconducting when the maximum temperature is below 10K. In order to satisfy this temperature requirement, cryogenic plants are essentially dedicated to this objective. For instance, the magnets of ITER will be cooled down by supercritical helium circulation at a temperature of 4.2 K. To do so, a "cold production system" (cryoplant), located on the ITER platform, will produce the necessary fluids and supply the installation through a complex network of "cold lines" (cryolines).

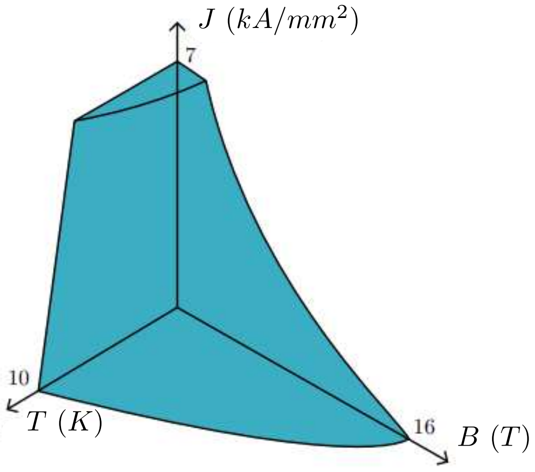


Figure 1.4: Critical surface of the niobium-titanium alloy expressed in the coordinate of current density J (in kA/m^2), temperature T (in K), and magnetic field B (in Tesla), under which the material has superconducting state.

In the next section, cryogenic refrigerators and the physical principles for generating liquid helium at atmospheric pressure are presented.

1.2 Helium Refrigerators

In order to cool superconducting magnets to a temperature at which gaseous helium liquefies, a combination of compression cycles and expansion cycles are employed in a cryogenic refrigerator. The sub-cycles that are implemented in a full refrigerator available at CEA, which has a cooling power of 400 W at 1.8 K, are presented in the sub-sections below.

1.2.1 Reversed Carnot cycle

It is absolutely possible to reverse the Carnot cycle where two reversible isothermal processes and two isentropic processes take place. Fig. 1.5 shows the synoptic view of the implementation and the temperature/specific entropy (T/s) diagram of the reversed Carnot cycle. The cycle consist of an isentropic expansion (4-1), an isothermal heat transfer (1-2), an isentropic compression (2-3) and an isothermal heat rejection (3-4). The efficiency coefficient of this cycle is defined by a ratio between the work Q_L needed for the system to extract a thermal work Q_H from a temperature T^H to T^C, namely:

$$\eta = \frac{T^C}{T^H - T^C}$$

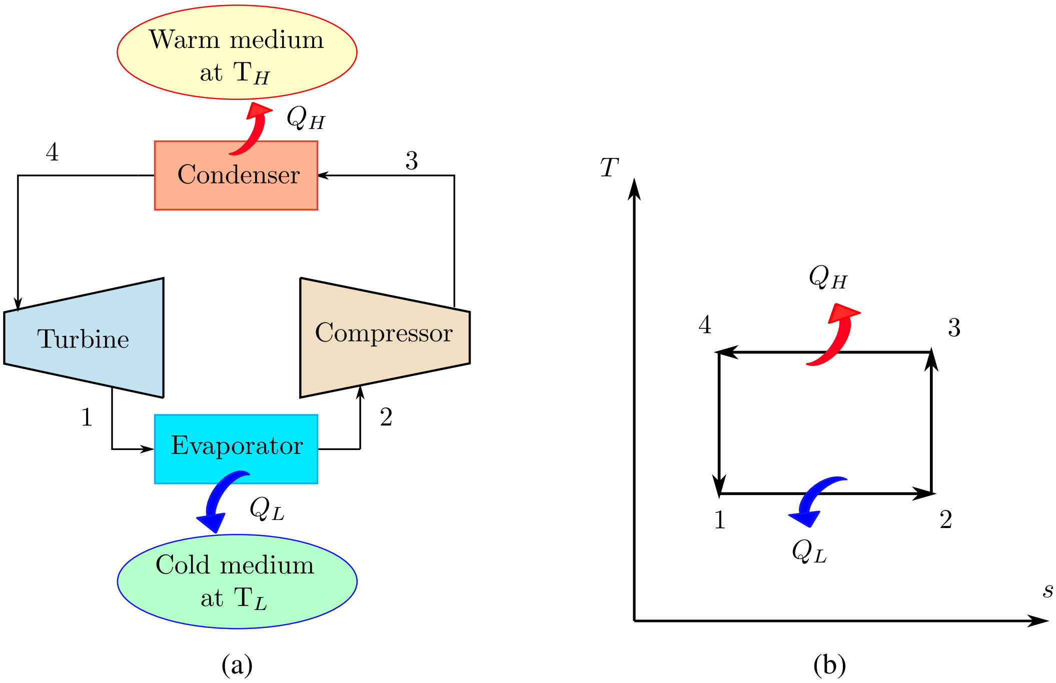


Figure 1.5: Synoptic view of Carnot cycle (a) and T/s diagram of Carnot cycle (b)

The reversed Carnot cycle is the most efficient refrigeration cycle operating between two specified temperature levels. However, this cycle cannot be implemented in practice because of the friction of the moving engines and the irreservabilities in the heat exchangers, turbines, compressors, etc. At present, the efficiency coefficient of the cryogenic refrigerator disposed at CEA is roughly 20% compared to the reversed Carnot cycle.

1.2.2 Joule Thomson cycle

The Joule-Thomson cycle is used to liquefy helium gas by expanding the gaseous fluid through a so-called Joule-Thomson (JT) valve. This cycle consists of a compression device, a heat exchanger between 1 and 2, and between 4 and 5. Joule-Thomson expansion occurs between 2 and 3 through the JT valve. Fig. 1.6 shows the synoptic view and the temperature-specific entropy (T/s) diagram of the cycle.

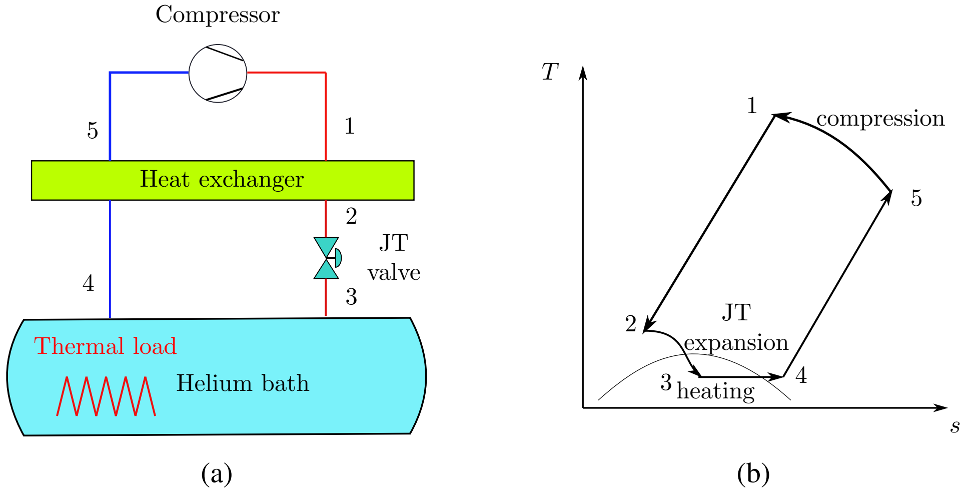


Figure 1.6: Synoptic view of Joule-Thomson cycle (a) and T/s diagram of Joule-Thomson cycle (b)

However, it is essential to note that the helium fluid is not sufficiently cooled down when passing through the JT valve. More precisely, if the helium gas is at a temperature higher than the inversion one (40K), the isenthalpic expansion will heat the fluid. In addition, being below this inversion temperature does not guarantee the production of liquid helium. For instance, an expansion from 20 bars to 1 bars of helium gas at 10 K only produces helium at 6.3 K, which is not in the liquid state. Hence, it is often to combine the Joule-Thomson cycle with some isentropic expansion cycles, which is presented in the following section.

1.2.3 Brayton cycle

In many large cryogenic refrigerators, several Brayton cycles can be placed in series to cool down the helium before it is liquefied through the JT valve. More precisely, the Brayton cycle consists of three components: a compressor, a turbine and a heat exchanger. Figure 1.7 shows the synoptic view of the Brayton cycle and the T/s diagram of the thermodynamic process it implements.

Compared to the Joule-Thomson cycle, the difference between these two cycles is the process that takes place between 2 and 3. Indeed, the passage of the fluid through the turbine causes the expansion process, which extracts energy from the fluid in the form of mechanical work and leads to a decrease in the fluid temperature.

The following section will describe the warm compression station, which is used to close the thermodynamic process of the cryogenic refrigerators.

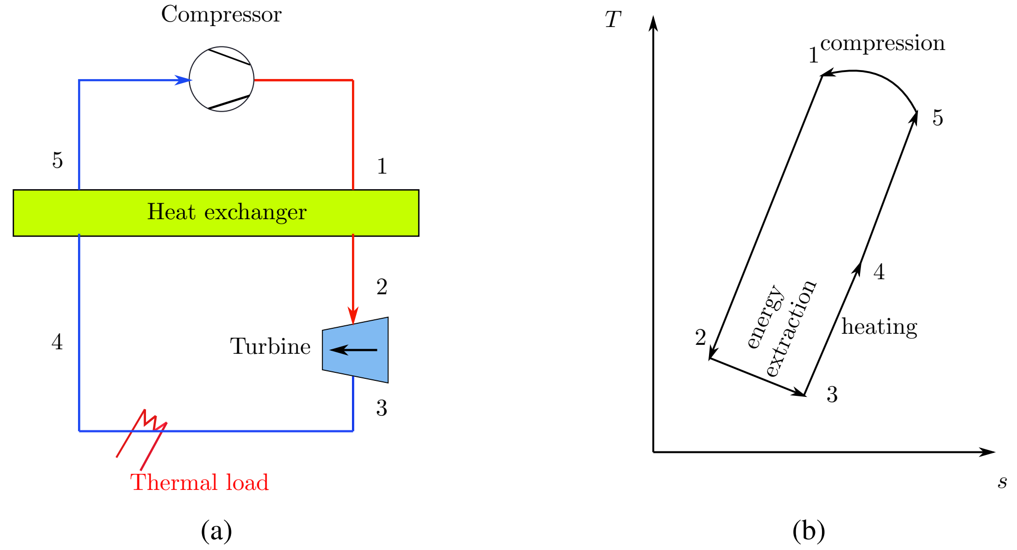


Figure 1.7: Synoptic view of Brayton cycle (a) and T/s diagram of Brayton cycle (b)

1.2.4 Warm compression station (WCS)

In cryogenic refrigerators, several cycles can be constructed in cascade order so that a certain cooling power can be reached. This is also the case of the 400 W refrigerator available at CEA, which will be presented in details in the next subsection. The operational pressures required by the cycles are created by one or a series of compressors that can be placed on top of the constructed sequence. Fig. 1.8 illustrates the WCS (without controlling valves) in the configuration using one compressor and two compressors. Typically, this plant maintains the helium gas pressure in two main lines, which are the high pressure pipeline and the low pressure pipeline denoted by P^H and P^L, respectively.

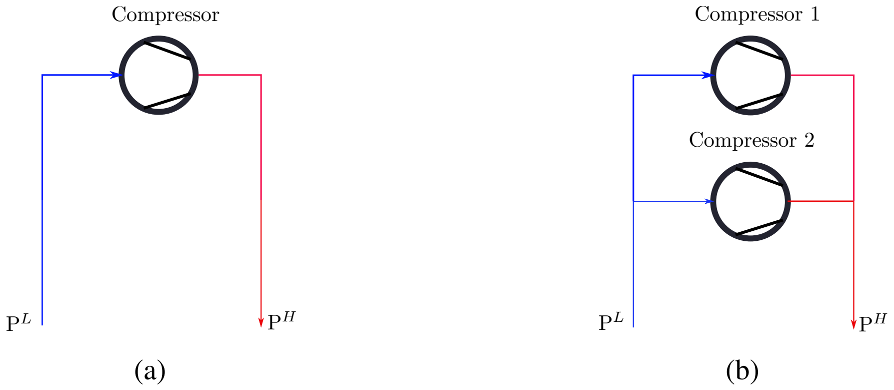
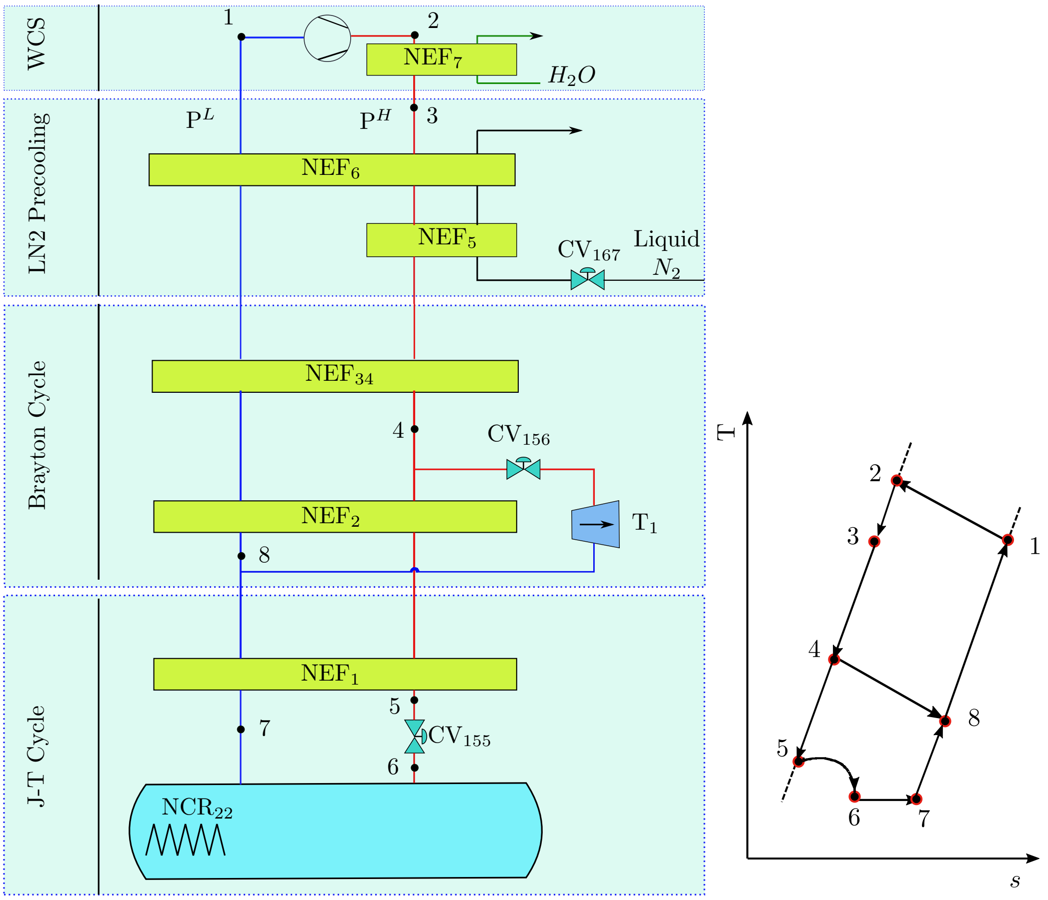


Figure 1.8: Synoptic views of the warm compression station using one compressor (a) and two compressors placed in parallel (b).

At this stage, all the elementary components have been introduced. The complete process that will be introduced in this study will be presented in the next section.

1.2.5 Station 400 W

The cryoplant available at CEA is introduced in this paragraph. This station consists of a Joule-Thomson cycle, a Brayton cycle, a pre-cooling stage with liquid nitrogen, and a warm compression station.



(a) (b)

Figure 1.9: Synoptic view of the 400 W @ 1.8 K station available at CEA (a) (presented in the 400 W @ 4.4 K configuration) and the associated T/s diagram (b)

In brief, this cryogenic refrigerator implements a closed thermodynamic cycle (Fig. 1.9b). In this cycle, the helium gas fluid flows clockwise through two main lines, namely the high pressure line (red line) and the low pressure line (blue line) shown in Fig. 1.9a. The cooling power of the cryogenic refrigerator is generated by heat exchange in the fluid through a series of heat exchangers denoted by NEF_x, and also by thermal energy extraction using a turbine denoted by T_1 (in the Brayton cycle). The helium gas is partially liquefied after passing through the valve denoted by CV_{155} and rests in the helium bath; the low-temperature gaseous part exits the bath through the cold pipe. The main objective is to reject the disturbing thermal power induced by the heat source designated by NCR_{22}. Finally, the cycle is closed by the so-called warm compression station, where the N$_c$ compressors reside. It should be noted that the synoptic view of the 400 W station presented in Fig. 1.9b is the configuration at 4.4 K, whereas the full configuration is at 1.8 K.

1.2.6 The pulsed thermal load

In the context of a tokamak application, the cooling power from a refrigerator is transferred to the tokamak via a distribution system (Fig. 1.10). During tokamak operation, the dynamics of the refrigerator are generally affected by pulsed charges from the tokamak magnets. The sources of these pulsed heat charges are numerous and can be listed as follows: currents flowing in the magnets (Joule effect in the magnet joints), AC losses, and neutron flux from the fusion plasma. For example, in the Japanese tokamak JT-60SA, the cooling system is expected to be affected by the pulsed load shown in Fig. 1.11.

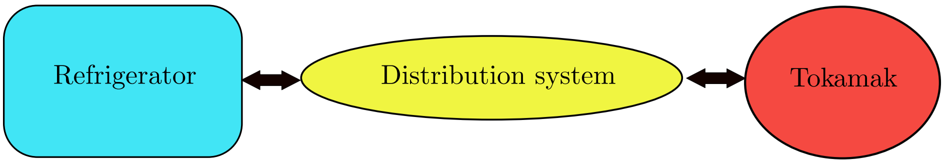


Figure 1.10: Synoptic view of the system of cooling power distribution.

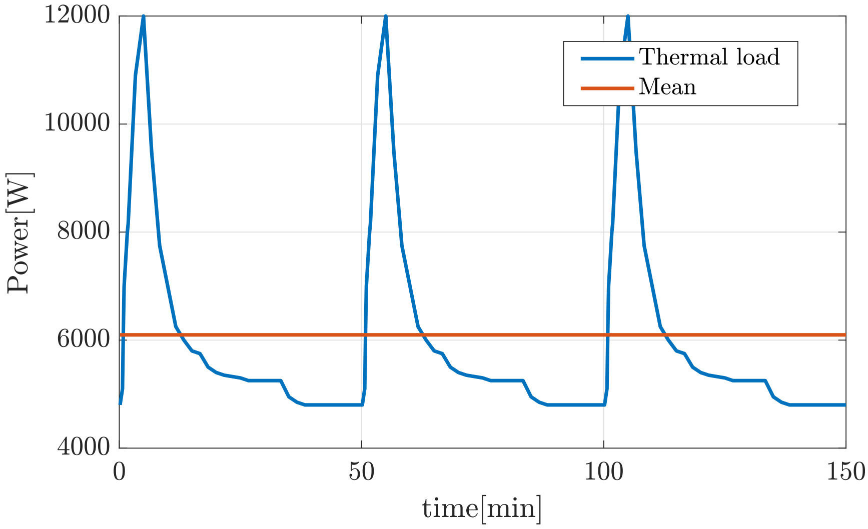


Figure 1.11: Pulsed charge of Tokamak JT60-SA [10].

It is noted that the maximum value of expected pulsed loads experienced by the JT-60SA tokamak refrigeration system reaches up to twice the average cooling capacity of the facility. This typical disturbance can cause system instability or even exceeds the system capacity (e.g.

exceed the compressor power).

1.3 Control design

In the previous section, the cryoplant used to cool some thermal loads was introduced. However, these refrigerators are typically built to handle near-constant loads over a long period of time. Many control methods have been developed and implemented to control the cold box and WCS, which can be found in [11, 12, 13] (for WCS control) and [14, 15]. (for the control of the cold box). In this section, some common methods that are widely used are presented.

1.3.1 Control of the WCS

Warm compression station control is not a new topic in cryogenic control engineering. Many effective approaches can be found in [11, 12, 13] . This paragraph will focus on some of the most widespread and advanced approaches that are used to control this station.

Recall that the WCS maintains the pressures of the incoming and outcoming fluids denoted by P^L and P^H in Fig. 1.12, respectively. To do so, a set of controlling valves is used, which are a bypass valve CV_{bp}, a discharge valve CV_{dch} and a charge valve CV_{ch}.

For controlling the WCS, the best known strategy used to maintain the pressure P^H of the outgoing fluid is the *split-range* [16] method combined with a PI controller to control the incoming fluid pressure P^L. More precisely, this method consists of two single-variable PI regulators, one for each pressure to be controlled (Fig. 1.12a). The first controller regulates the low pressure P^L via the bypass valve CV_{bp}. For the high pressure P^H, the valves CV_{dch} and CV_{ch} are considered as a single actuator that is driven by the PI controller associated with the high pressure P^H. If the high pressure is too high, the discharge valve opens to remove gas from the circuit, while on the contrary, if the high pressure is too low, the charge valve opens to add gas to the circuit. The second method consists [17] of two multivariable LQ controllers, each of which controls two actuators, either Cv_{bp} and Cv_{dch}, or Cv_{bp} and Cv_{ch}, to control the high pressure and low pressure, respectively (Fig. 1.12b). Since only two scenarios can occur, either a lack of gas or an excess of gas in the system, one of the two controls for these regulators is selected. The principle of selection is based on the fact that the allowed controller is the one that gives a positive value on the discharge valve or the charge valve.

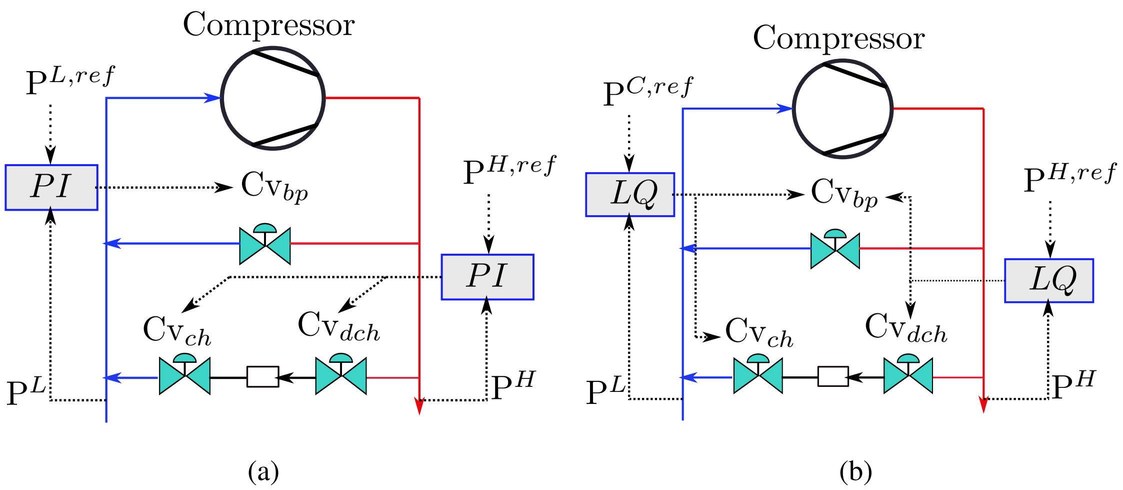


Figure 1.12: Synoptic view of the warm compression station with the associated PI controllers (a) and the LQ controllers (b).

1.3.2 Control of the Cold-box

The cold box combines two subsystems which are the Joule-Thomson cycle and the Brayton cycle (Fig. 1.9a). The controlled outputs of this plant are the liquid helium level in the bath Ltb_{131}, the temperature at the inlet of the valve J-T Ttb_{108} and the temperature at the outlet of the turbine Ttb_{130}. The manipulated input are the control valve CV_{155}, the heating power $\mathrm{NCR}_{22}^{(a)}$ in the J-T cycle and the control valve CV_{156} in the Brayton cycle. Note that this plant is subjected under a thermal disturbance $\mathrm{NCR}_{22}^{(w)}$, which then contributes together with the manipulated power $\mathrm{NCR}_{22}^{(a)}$ to the total heating power, namely, $\mathrm{NCR}_{22} = \mathrm{NCR}_{22}^{(a)} + \mathrm{NCR}_{22}^{(w)}$.

Many methods are used to control the cold box: helium level control by heating actuator, variable high pressure control, etc. This paragraph will present the PI controller approach and the model predictive control approach that have been implemented on the 400 W station at CEA.

In order to control the cold box of the refrigerator, the simplest way is to use several PI type single-variable controllers. When the steady state of the system is established, the control principle can be stated as follows: the temperature at the inlet of the JT valve is controlled using a turbine. The output temperature of this turbine is controlled by a PI controller. Once the inlet temperature of the JT valve is stable, the cooling power of the cryoplant is then constant and always higher than the thermal load. Thus, a heating actuator is added in the helium bath which is driven by another PI controller to regulate the level of liquid helium in the bath.

In [18], a constrained model predictive controller is used to control the cold box. However, the computational burden associated with this type of controller makes it impractical in real-time implementation, especially when the control problem becomes more complex due to the increase of states and constraints to be considered. Therefore, the authors have proposed a real-time solver that can be implemented in a programmable logic controller (PLC). In short, this solver implements a gradient-based method to find the minimum of a quadratic function. The efficiency of this solver has also been evaluated with different control updating periods, demonstrating that the more frequently the control input is updated, the more efficient the closed-loop performance obtained.

1.4 Context and problem statement

After discovering some of the methods used to control the cryogenic subsystems, these techniques can be separated into two categories: centralized control and decentralized control. For instance, the system under a decentralized control structure is assumed to have several interacting subsystems, each of which is controlled by a local controller. However, strong interaction between subsystems can prevent the whole process from achieving stability/performance. On the other hand, as long as MPC is involved, the approach that uses only one particular controller for a global system is a centralized approach since it tries to solve a large-scale problem in which the interactions between subsystems are considered.

Although centralized approaches with MPC have been used widely in real-life applications, they also have many drawbacks that are listed below:

Problem 1: Solving the centralized problem
Solving the optimal control problem for a large-scale system at each constrained update period is an extreme challenge. Indeed, the complexity of the control problem to be solved increases with the complexity of the system, as does the relatively necessary computation. Moreover, the computational resources available on the platform are not always adapted to the complexity of the calculations, which prevents the control algorithm to be deployed.

Problem 2: Nonlinearity isolating
As soon as some nonlinear characteristics of a single subsystem is to be considered, the centralized MPC problem is obliged to be treated as a nonlinear one while the other subsystems could be considered linear. Thus, this nonlinear subsystem can be considered the bottleneck for any centralized framework as it makes modeling, control synthesis, and implementation more difficult. Hence, the need of a control framework that allows to isolate the nonlinear subsystem from the linear ones, while taking into account the coupling effects between them is absolutely an emerging topic.

Problem 3: Flexibility of the control algorithm
In general, large-scale systems usually experience changes in their life cycle. These changes

could be controller adjustment, actuator replacement, etc. Especially when centralized control is employed, any change in local subsystems requires the overall performance to be assessed.

Problem 4: Different time scales
A large-scale system may be composed of several subsystems, among which different time scales are encountered. Indeed, many systems are characterized by clearly distinguishable slow and fast dynamics. For instance, the WCS is actively controlled with a sampling period of 1 second, while the cold box's sampling period of 5 seconds due to its slow dynamic behavior. If the centralized approach is chosen, the targeted model should be based on the faster sampling time constant, while the prediction horizon is chosen long enough so that the stability of the slow dynamic subsystem is guaranteed. Consequently, the control problem might not be successfully solved in that short period in order to control the system efficiently.

Facing to these problems, this book tries to develop and implement a hierarchical control approach that ensures stability and performance of the global system.

1.5 Book outline

This section briefly summarizes the main content of this book, chapter by chapter, which are listed below:

- Chapter 2: *Background*

  The essential preliminaries to our main research are presented, which is the hierarchical control approach in cryogenic processes. First, the decomposition method that is required to separate the overall system into many subsystems interacting with each other via coupling signals is introduced. Since the topic of this book is to propose a control method for large-scale systems, the general observer-based decentralized control framework is described, which is served as a baseline to be compared with the proposed method throughout this book. The local controllers of the subsystems in the framework can be chosen to be the well-known and widely used model predictive control, whose formulation is then recalled. Finally, the centralized observation scheme for state estimation is described.

- Chapter 3: *Fixed-point-iteration-based hierarchical control*

  We present the fixed-point-based hierarchical control framework, which consists of two distinct layers. At the lower layer, interconnected subsystems are controlled by local agents, which require coupling profiles to compute their control profiles. At the top layer, the coordinator communicates extensively with the agents to find the true coupling profiles associated with a given setpoint. In addition, the coordinator optimizes a central cost with respect to the setpoints that will be sent to the agents. In the original work [1], this framework was validated in the case where local agents implement only unconstrained linear controllers. We extended this work by replacing these controllers with constrained nonlinear controllers and evaluating the effectiveness of the framework in this case. In addition, the real-time implementation problem resulting from the use of nonlinear controllers is also addressed by proposing a technique that consists of distributing the optimization over time. Finally, a series of simulations show the effectiveness of the proposed framework.

- Chapter 4: *On the use of fast-NMPC and deep learning approach in fixed-point-iteration-based hierarchical control*

  In this framework, the constrained nonlinear MPC problems are solved repeatedly until the communication between the coordinator and the local agent converges to a fixed point, which definitely increases the computational load. Indeed, there will be a bottleneck for real-time implementation if the required computation time is far from the update period. In order to reduce the computation time of local controllers, fast nonlinear MPC techniques such as truncated gradient solver and deep-neural-network-based controller are proposed. It is also shown that by doing so, the control update period can be significantly reduced, and the closed-loop performance greatly improved. This chapter can thus be seen as a concrete implementation and validation of some key ideas in the design of real-time distributed NMPCs.

- Chapter 5: *Application to a complete refrigerator and some development in the fixed-point based hierarchical control method*

  The proposed framework is applied to the complete system, which consists of eight subsystems with different time scales. A residual-based iterative method, which is called Anderson acceleration method, is implemented to ensure convergence the inner loop processed between the coordinator and the agents. A gradient-based solver is implemented at the coordination level to optimize the central cost with respect to the set point. Numerical simulations are performed to evaluate the effectiveness of the proposed framework.

CHAPTER 2

Background

Abstract In this chapter, we present the tools used for modeling, control law synthesis and dynamic simulation of cryoplants. In the context of decentralized control of cryogenic processes, a particular way to decompose the overall system is presented. Then, a decentralized control schema is introduced, which will serves as a comparison base to the hierarchical control framework presented in chapter 3. In addition, the MPC formulations which are implemented to control the subsystems are also recalled. Finally, the observation method that is used in both decentralized and hierarchical methods is described.

2.1 Modeling Tool

For system modeling and simulation, there are many tools, both homemade and commercial versions, which are now available on the market, such as Cryolib [19], AspenHysys Dynamics [20], GT-SUITE [21], etc. However, these tools are dedicated for simulating cryogenic installations' dynamic but not for control synthesis, the details are given below:

- With AspenHysys Dynamics, the derived model is very complex, deduced from the fact that, according to the author, it takes two hours of calculation to simulate fifteen hours of operation of the machine. This is due to the heat exchangers, which are modeled in a very precise way with many differential equations. This model is not simple enough to deduce a control law.
- The software Cryolib allows to modelize complexe cryogenic plants. It has been validated by CERN (Switzerland) and has been playing an important role in many applications. However, the complexity and the form of writing of the final model does not make it compatible with the generation of control laws.

Recognizing the urgent need for a tool dedicated to the synthesis and validation of controls specific to cryogenic systems, a library named Simcryogenics[22] was developed at CEA (France).

Simcryogenics is a library developed within the Matlab/Simulink/Simscape environment (Fig. 2.1). This library provides many standard cryogenic components such as: heat ex-

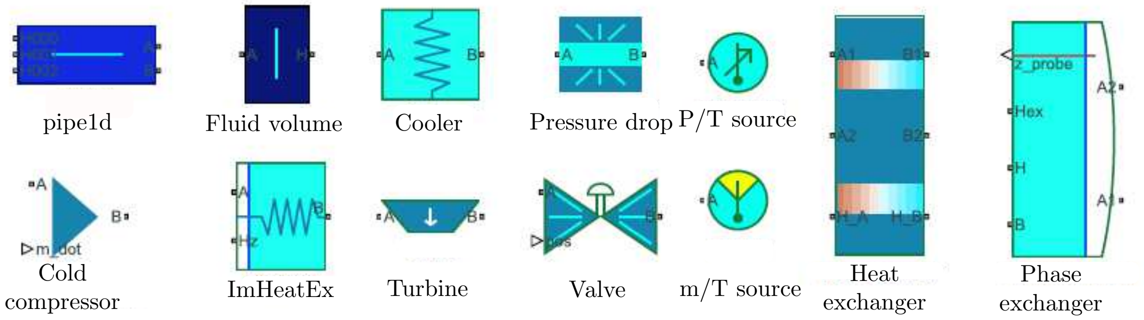


Figure 2.1: Example of components available in the Simcryogenics library [22]

changer, valve, compressor, phase separator, etc. Those components can be dragged and dropped on a Simulink worksheet and connected together according to certain rules, to create a complete functioning model.

The models generated by this library are not only simple enough but also ensure accuracy in system dynamics; any linear control law can be synthesized directly from the linear dynamic equations obtained by using the Matlab *linearize()* subroutine combined with the model created by this library.

All of the performed results showed in this book are obtained by using this library.

2.2 System decomposition

This book aims to present a hierarchical control approach, in which the system is controlled by several local agents. It is therefore necessary to present the method used to decompose the complete system into smaller subsystems that interact with each other.

Let's take as an example a group of two cycles, which are the Joule-Thomson (J-T) cycle and the Brayton cycle of the 400 W refrigerator described in 1.3.2. This system can be decomposed in two possible ways: either by cycles or by elementary components. Fig. 2.2 shows the mentioned decompositions for this group of cryogenic cycles. More precisely, in the 2-subsystem topology, the group is separated into its two natural cycles, which are the J-T cycle and the Brayton cycle. In the 4-subsystem topology, except for the J-T cycle, the Brayton cycle is separated into 3 elementary components which are NEF_2, NEF_{34} and the combination of the turbine T_1 and the valve Cv_{156}.

Intentionally, these decompositions attempt to separate the large system into a network of smaller subsystems. In this network, the subsystems interact mutually with each other via the coupling signals $v_{s \to s'}$, which represents the output of subsystem $S - s$ that affects the

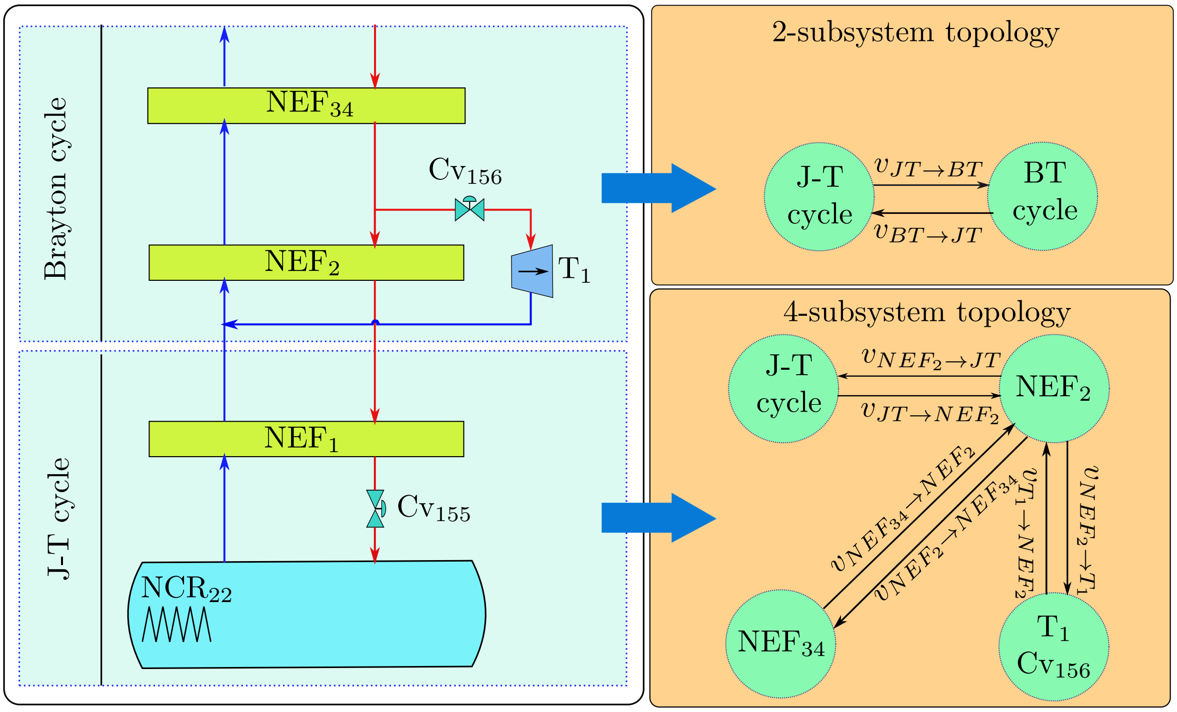


Figure 2.2: Two possible decompositions of the coldbox of the 400 W station.

dynamic of subsystem $S_{s'}$.

In thermodynamics, the state of a fluid can be determined using 2 variables, namely, the specific enthalpy and pressure. As cryogenic processes always operate with forced flows, the flow rate is also to be taken in to account. However, as long as the phase of the fluid is monophasic (in our case liquid or gaz) the specific enthalpy can be replaced by the temperature as the information to determine the state of the fluid. Based on this, the coupling signals of each subsystem are determined as follows:

- Counting from the helium bath, the upper heat exchangers impose pressures on their lower neighbors, while the lower heat exchangers impose flow rates on their upper neighbors.
- Meanwhile, the origin subsystem of the temperature is identified by the fluid direction (the arrow line in the system scheme of Fig. 2.2); it belongs to the subsystem from which the fluid exits.
- Finally, the actuators such as turbines and valves impose their outlet temperatures and flow rates on the components connected to them.

Note that in the 2-subsystem and 4-subsystem topologies mentioned above, all the decoupling positions are where the fluid is purely gaseous. Thus, the specific enthalpies could be replaced by the temperatures.

Table 2.1: The associated coupling vectors of 2-subsystems topology and 4-subsystems topology.

2-subsystem topology	
Subsystem s	$v_{s\to s'}$
J-T cycle	$v_{JT\to BT} = [M^H_{JT\to BT}, M^L_{JT\to BT}, T^L_{JT\to BT}]^T$
Brayton cycle	$v_{BT\to JT} = [P^H_{BT\to JT}, P^L_{BT\to JT}, T^H_{BT\to JT}]^T$
4-subsystem topology	
J-T cycle	$v_{JT\to NEF_2} = [M^H_{JT\to NEF_2}, M^L_{JT\to}]^T$
NEF$_2$	$v_{NEF_2\to JT} = [P^H_{NEF_2\to JT}, P^L_{NEF_2\to JT}, T^H_{NEF_2\to JT}]^T$ $v_{NEF_2\to NEF_{34}} = [M^H_{NEF_2\to NEF_{34}}, M^L_{NEF_2\to NEF_{34}}, T^L_{NEF_2\to NEF_{34}}]^T$ $v_{NEF_2\to T_1} = P^L_{NEF_2\to T_1}$
NEF$_{34}$	$v_{NEF_{34}\to NEF_2} = [P^H_{NEF_{34}\to NEF_2}, P^L_{NEF_{34}\to NEF_2}, T^H_{NEF_{34}\to NEF_2}]^T$ $v_{NEF_{34}\to T_1} = [P^H_{NEF_{34}\to T_1}, T^H_{NEF_{34}\to T_1}]^T$
T$_1$	$v_{T_1\to NEF_{34}} = M^H_{T_1\to NEF_{34}}$ $v_{T_1\to NEF_2} = [M^L_{T_1\to NEF_2}, T^L_{T_1\to NEF_2}]^T$

To conclude, Tab. 2.1 summarizes the coupling signals associated to each decomposition. The symbols $T^H_{s\to s'}$, $P^H_{s\to s'}$ and $M^H_{s\to s'}$ / $T^L_{s\to s'}$, $P^L_{s\to s'}$ and $M^L_{s\to s'}$ represent the temperature, the pressure and the flow rate at the high pressure / low pressure pipeline through which the subsystem s affects the subsystem s'.

2.3 Decentralized control and MPC

2.3.1 Decentralized control approach

In this book, we will focus on the decentralized approach and hierarchical approach to control large-scale systems. These two approaches are eventually compared to each other to evaluate their performances. First, the decentralized approach is presented in this section, while the targeted hierarchical control method that is based on fixed-point iteration is presented in chapter 3.

Before going any further, some notations that are extensively used in the sequel need to be defined.

Notation For a sequence of vectors $q_{i_1}, q_{i_2}, \dots, q_{i_n}$, the stacked vector q that concatenates this sequence elements is defined as follows:

$$q := \underset{i\in\mathcal{I}}{\operatorname{col}}(q_i) = [q_{i_1}^T, \dots, q_{i_n}^T]^T, \text{ with } \mathcal{I} := \{i_1, \dots, i_n | i_1 < \dots < i_n\} \tag{2.1}$$

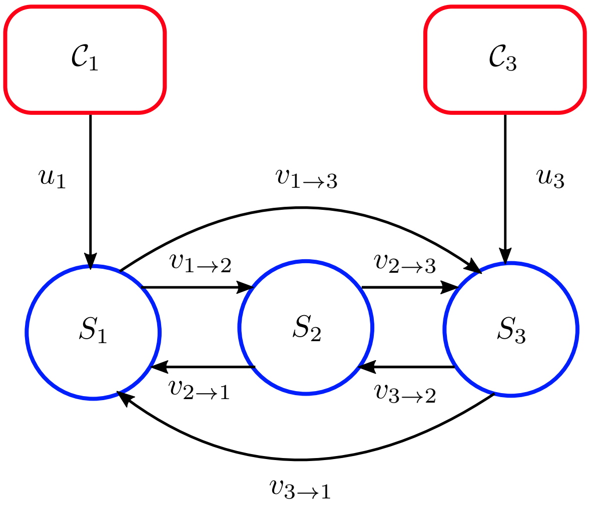


Figure 2.3: Example of decentralized control structure. The instantiation of the sets corresponding to this scenario are: $\mathcal{N} = \{1,2,3\}$, $\mathcal{N}^{ctr} = \{1,3\}$, $\mathcal{N}^{unc} = \{2\}$, $\mathcal{N}_1 = \{2,3\}$, $\mathcal{N}_2 = \{1,3\}$ and $\mathcal{N}_3 = \{1,2\}$.

where $\text{col}_{i\in\mathcal{I}}(q_i)$ is the concatenation operator. The bold-faced notation $\boldsymbol{p}$ denotes the profile of a vector variable p over a prediction horizon of length N, namely:

$$\boldsymbol{p} := [p^T(k),\dots,p^T(k+N-1)]^T \in \mathbb{R}^{N\cdot n_p} \tag{2.2}$$

Given a network of n_s subsystems, the set $\mathcal{N} := \{1,\dots,n_s\}$ gathers all subsystem indices, which is divided into two subsets $\mathcal{N}^{unc}$ and $\mathcal{N}^{ctr}$. The indices that respectively belong to the subset $\mathcal{N}^{ctr}$ / $\mathcal{N}^{unc}$ refers to the subsystems that have / do not have control input, respectively. Each of the controlled subsystem $S_{s\in\mathcal{N}^{ctr}}$ has a controller denoted by $C_{s\in\mathcal{N}^{ctr}}$. Furthermore, the subsystem S_s affects the dynamic of the subsystems $S_{s'}$ through its output $v_{s\to s'}$, for $s' \in \mathcal{N}_s$ with $\mathcal{N}_s$ denoting the set of neighbors of subsystem S_s. As an illustration, Fig. 2.3 shows a synoptic view of a typical decentralized architecture applied to an interconnecting network of subsystems.

The mathematical models of the subsystems are given below:

- For all subsystems S_s ($s \in \mathcal{N}$), the dynamic model is given by:

$$x_s^+ = f_s(x_s, u_s, w_s, v_s^{in}) \tag{2.3}$$

  where $x_s \in \mathbb{R}^{n_x^{(s)}}$, $u_s \in \mathbb{R}^{n_u^{(s)}}$ and $w_s \in \mathbb{R}^{n_w^{(s)}}$ represent respectively the state vector, the manipulated input and the disturbance input of the subsystem S_s. Note that for uncontrolled subsystems $S_{s\in\mathcal{N}_{unc}}$ and subsystems that are not affected by any disturbance, their control input u_s and disturbance input w_s do not exist in the above model equations, i.e., $u_s = \varnothing$ or $w_s = \varnothing$. Furthermore, the coupling input v_s^{in} concatenates all the coupling signals that affect the dynamics of subsystem S_s, which is $v_s^{in} = \text{col}_{s'|s\in\mathcal{N}_{s'}}(v_{s'\to s})$.

- The outgoing coupling signal $v_{s\rightarrow s'}$ that affects $S_{s'}$ is described by:

$$v_{s\rightarrow s'} = g_{s\rightarrow s'}(x_s, u_s, v_s^{in}) \quad \text{for } s' \in \{s' | s \in \mathcal{N}_{s'}\} \tag{2.4}$$

  with $v_{s\rightarrow s'} \in \mathbb{R}^{n_v^{(s\rightarrow s')}}$. Note that these variables can be gathered to form a vector representing the outgoing coupling signals coming from S_s denoted by:

$$v_s^{out} = \underset{s'\in\mathcal{N}_s}{\text{col}} (v_{s\rightarrow s'}) = g_s^{out}(x_s, u_s, v_s^{in}) \tag{2.5}$$

- The output to be controlled is given by:

$$y_s = h_s(x_s, u_s, v_s^{in}) \tag{2.6}$$

  with $y_s \in \mathbb{R}^{n_y^{(s)}}$.

Decentralized control frameworks consist of having several local controllers (agents) $C_{s\in\mathcal{N}^{ctr}}$ for the separable subsystems (Fig. 2.3). These agents operate independently of each other, which means there is no communication between them, neither with any master level. The design and the success of these local controllers is trivial when the coupling signals are weak. However, it has been shown that the stability and / or good performance with a decentralized framework may not be achievable due to strong interactions between the subsystems [23, 24].

For literature reviews, reader is referred to classical textbooks [25, 26] on decentralized control structure and stability. In addition, some methods based on vector Lyapunov functions [26], sequential design, specific decomposition or optimization are also available [27, 28, 29]. In the context of decentralized control, the survey papers such as [30, 31, 32] have recently reported an up-to-date list of references.

In large-scale industrial applications, these local controllers can be of different types (e.g., PID, LQR, MPC, ...). With well-known controllers such as the PID type controller, the P, I and D coefficients need to be carefully adjusted to handle the strong coupling between subsystems. On the other hand, linear-quadratic regulator (LQR) and MPC are more and more popular nowadays because of their intrinsic multivariable nature. Furthermore, nonlinearity and operating constraints can also be included in the optimization control problems of MPC formulations. This is the reason why MPC formulations is recalled in the next section.

2.3.2 Model predictive control

2.3.2.1 MPC formulation

In the process control industry, model predictive control has been successfully applied in the last two decades. Its popularity is due to the ability to incorporate the actual control objective and operating constraints in the optimization control problem solved at each sampling time.

Given an initial state vector $x_s(k)$, an incoming coupling profile $\boldsymbol{v}_s^{in}$, a disturbance profile $\boldsymbol{w}_s$ and any control profile $\boldsymbol{u}_s$ defined over the prediction horizon $[k, k+N]$, the corresponding nominal state trajectory is given by:

$$x_s(k+i+1) = f_s(x_s(k+i), u_s(k+i), v_s^{in}(k+i), \boldsymbol{w}_s(k+i)) \quad \text{for}\, i = 0, \dots, N-1 \tag{2.7}$$

The state profile $\boldsymbol{x}_s$ over a horizon of length N, can be defined by a straight-forward notation:

$$\boldsymbol{x}_s = \mathbf{f}_s(x_s(k), \boldsymbol{u}_s, \boldsymbol{v}_s^{in}, \boldsymbol{w}_s); \tag{2.8}$$

Similarly, the output profile can be computed by using (2.6), namely:

$$\boldsymbol{y}_s(i) = h_s(\boldsymbol{x}_s(i), \boldsymbol{u}_s(i), \boldsymbol{v}_s^{in}(i)); \quad \text{for}\, i = 0, \dots, N-1 \tag{2.9}$$

where $\boldsymbol{x}_s(i)$ refers to the prediction of state value in time-domain that is $x_s(k+i)$, and for the sake of brevity it can also be written in the following form:

$$\boldsymbol{y}_s = \mathbf{h}_s(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{v}_s^{in}); \tag{2.10}$$

Besides, the outgoing coupling profile can be deduced as follows:

$$\boldsymbol{v}_s^{out} = \mathbf{g}_s^{out}(\boldsymbol{x}_s, \boldsymbol{u}_s, \boldsymbol{v}_s^{in}); \tag{2.11}$$

At every instant k, the standard formulation of the optimal control problem are given by:

$$\begin{aligned}
\mathcal{P}_s : \quad & \min_{\boldsymbol{u}_s} \quad J_{\text{MPC}}^{(s)}(x_s(k), \boldsymbol{u}_s, \boldsymbol{v}_s^{in}, \boldsymbol{w}_s, r_s) && (2.12)\\
& = \sum_{i=0}^{N-1} \|x_s^{sp} - x_s(k+i)\|_{Q_s}^2 + \|u_s^{sp} - u_s(k+i)\|_{R_s}^2 && (2.13)\\
& \text{subject to :} && \\
& \boldsymbol{x}_s = \mathbf{f}_s(x_s(k), \boldsymbol{u}_s, \boldsymbol{v}_s^{in}, \boldsymbol{w}_s) && (2.14)\\
& \boldsymbol{u}_s \in \mathbb{U}_s && (2.15)
\end{aligned}$$

where $Q_s \in \mathbb{R}^{n_x^{(s)} \times n_x^{(s)}}$ and $R_s \in \mathbb{R}^{n_u^{(s)} \times n_u^{(s)}}$ are weighting matrices on the state and control input. In the decentralized approach, the incoming coupling profile $\boldsymbol{v}_s^{in}$ is considered constant over the prediction horizon and equal to the estimated value given by the observer (see later). Similarly, the disturbance profile $\boldsymbol{w}_s$ (if any) has a constant value over the prediction horizon, which is equal to the current disturbance value $w_s(k)$.

The stationary state and input denoted respectively by x_s^{sp} and u_s^{sp} depend on the output set-point r_s and are computed by solving the following optimization problem:

$$\begin{aligned}
\mathcal{P}_s^{sp} : \quad & \min_{x_s^{sp}, u_s^{sp}} \|y_s^{sp} - r_s\|_{Q_s^{sp}}^2 + \|u_s^{sp}\|_{R_s^{sp}} && (2.16)\\
& \text{subject to :} && \\
& y_s^{sp} = h_s(x_s^{sp}, u_s^{sp}, v_s^{end}) && (2.17)\\
& x_s^{sp} = f_s(x_s^{sp}, u_s^{sp}, w_s, v_s^{end}) && (2.18)
\end{aligned}$$

where $v_s^{end} := \boldsymbol{v}_s^{in}(N-1)$ is the last vector of the incoming coupling profile $\boldsymbol{v}_s^{in}$, whereas $Q_s^{sp} \in \mathbb{R}^{n_y^{(s)} \times n_y^{(s)}}$, $R_s^{sp} \in \mathbb{R}^{n_u^{(s)} \times n_u^{(s)}}$ are respectively the weighting matrices on output and input. Note that (2.18) is the stationary condition associated to the set-point r_s.

In practice, when the length of the prediction horizon N is long enough for stability to be achieved, the resulting computational burden and memory footprint can be extremely challenging for real-time implementation. Therefore, it is preferable to parameterize the vector of decision variables $\boldsymbol{u}_s$ in the problem (2.13) to reduce the actual complexity of the problem and thus reduce the computational time and memory requirements. This technique will be introduced hereafter.

2.3.2.2 Parametrization

The parametrization consists in transforming the control profile by using a representation with a lower degrees of freedom. Particularly, this is often obtained by approximating every control decision in the full control sequence with a piece-wise constant function. More precisely, the values of the elements in the control profile between two predefined decision instants are defined by linear interpolation.

More precisely, given a vector denoted by $I \in \mathbb{N}^{n_I}$ that defines n_I decision instants in the prediction horizon of length N, namely:

$$I = [I_1, I_2, \ldots, I_{n_I}] \tag{2.19}$$

The parametrized vector $\boldsymbol{u}_s \in \mathbb{R}^{n_u \cdot N}$ can be obtained by the following formula.

$$\boldsymbol{u}_s = \Pi_s^I \cdot \boldsymbol{u}_s^p \tag{2.20}$$

where $\boldsymbol{u}_s^p$ is the new vector of decision variables and Π_s^I is the parametrization matrix given by:

$$\Pi_s^I = (a_{ij}) \text{ such that } a_{ij} = \begin{cases} \mathbb{I}^{n_u} & \text{if } I_j = i \\ \mathbb{I}^{n_u} \dfrac{I_{j+1} - i}{I_{j+1} - I_j} & \text{if } j < n_I \text{ and } I_{j+1} > i > I_j \\ \mathbb{I}^{n_u} \dfrac{i - I_{j-1}}{I_j - I_{j-1}} & \text{if } j > 1 \text{ and } I_j > i > I_{j-1} \\ 0 & \text{otherwise} \end{cases} \tag{2.21}$$

$\forall i \in [i, N], \forall j \in [1, n_I]$.

For instance, given the vector $I = [1\ 3\ 5\ 10\ 20]$ whose elements indicate the freedom instants in the prediction horizon of length $N = 20$ and a parametrized control profile $\boldsymbol{u}^p = [22\ 25\ 27\ 30\ 28]$ (for the purpose of illustration), the full control profile can be computed by using (2.20). Fig. 2.4 illustrates the control profile $\boldsymbol{u}$ deduced from the given parametrized vector $\boldsymbol{u}^p$. .

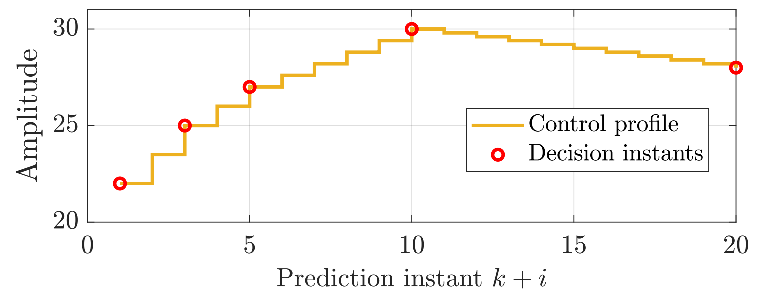


Figure 2.4: Example of a parameterization with linear interpolation.

It is important to note that the disturbance profile and incoming coupling profile can be similarly parametrized. Indeed, by defining the decision instants V and W, the resulted parametrized disturbance and incoming coupling profiles are given below:

$$\boldsymbol{w}_s = \Pi_s^W \cdot \boldsymbol{w}_s^p \quad \text{and} \quad \boldsymbol{v}_s^{in} = \Pi_s^V \cdot \boldsymbol{v}_s^{in,p} \tag{2.22}$$

Then, the MPC problem stated in (2.16)-(2.18) becomes:

$$\boldsymbol{u}_s^{p,opt} = \underset{\boldsymbol{u}_s^p}{\text{argmin}} \quad J_{\text{MPC}}^{(s),p}(x_s(k), \boldsymbol{u}_s^p, \boldsymbol{v}_s^{in,p}, \boldsymbol{w}_s^p, r_s) \tag{2.23}$$

$$\text{subject to:} \quad \boldsymbol{x}_s = \mathbf{f}_s^p(x_s(k), \boldsymbol{u}_s^p, \boldsymbol{v}_s^{in,p}, \boldsymbol{w}_s^p) \tag{2.24}$$

$$\boldsymbol{y}_s = \mathbf{h}_s^p(\boldsymbol{x}_s, \boldsymbol{u}_s^p, \boldsymbol{v}_s^{in,p}, \boldsymbol{w}_s^p) \tag{2.25}$$

$$\boldsymbol{u}_s^p \in \mathbb{U}_s \tag{2.26}$$

Consequently, the optimal control profile can be deduced by:

$$\boldsymbol{u}_s^{opt} = \Pi_s^I \cdot \boldsymbol{u}_s^{p,opt} \tag{2.27}$$

Once the optimal control profile $\boldsymbol{u}_s^{opt}$ is computed, the first control action, namely:

$$u_s(k) := [\mathbb{I}_{n_u^{(s)}}, \mathbb{O}_{n_u^{(s)}}, \ldots, \mathbb{O}_{n_u^{(s)}}] \cdot \boldsymbol{u}_s^{opt} \tag{2.28}$$

with $\mathbb{I}_{n_u^{(s)}}$ and $\mathbb{O}_{n_u^{(s)}}$ being respectively the identity and zero matrices of dimensions $n_u^{(s)} \times n_u^{(s)}$, is applied to the system during the updating period $[k, k+1]$.

It is essential to note that the local agents can also implement any type of regulators. Indeed, given the current state $x_s(k)$, the control sequence $\boldsymbol{u}_s$ over a prediction horizon of length N of any type of controller other than MPC can also be computed by simulating the corresponding controlled system's dynamic.

2.4 State estimation

As far as model-based control methods are concerned, system states are necessary for updating the control feedback. It should be recalled that the main objective is to have a complete decentralized control scheme in which observation and control are designed individually for each local subsystem. However, it is challenging to design such a framework specific to the applications interested in this book. This is due to the fact that the subsystems are highly coupled to each other and some of them do not have any available measurement to update correctly their states. Therefore, in order to prevent the observation problem from becoming the bottleneck of the whole idea, we decided to use the centralized observation framework, in which a single observer is designed to estimate the states of the whole system and allocate them to the corresponding subsystems. The reasons for choosing this solution are explained as follows:

- The design of the centralized observer remains independent of the local controller as it is based on the pair (u, y) where u results from whatever controller inside each subsystem and y is the measurement of output. In other words, the design of the observer remains valid even if a change in the control design is operated.
- Designing a complete decentralized observer might be technically impossible because of the available sensors at each local level.
- The proposed mixed scheme (hierarchical/decentralized for control and centralized for the observer) remains more realistic than many works on decentralized or hierarchical designs where the observation problem is completely ignored and a state measurement is supposed to hold.

First, a simple centralized linear observer is introduced. By linearizing the total plant, the linear dynamic model [disturbance-free] is obtained, namely:

$$\tilde{x}^+ = A \cdot \tilde{x} + B \cdot \tilde{u} \tag{2.29}$$

$$\tilde{y} = C \cdot \tilde{x} \tag{2.30}$$

where $\tilde{x} = \underset{s \in \mathcal{N}}{\operatorname{col}}(\tilde{x}_s)$, $\tilde{u} = \underset{s \in \mathcal{N}^{ctr}}{\operatorname{col}}(\tilde{u}_s)$ and $\tilde{y} = \underset{s \in \mathcal{N}^{ctr}}{\operatorname{col}}(\tilde{y}_s)$ are vectors that group the deviations of states, inputs and outputs from their operation values x_s^{op}, u_s^{op} and y_s^{op} of all the subsystems. Whereas, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$ and $C \in \mathbb{R}^{n_y \times n_x}$ are linearized matrices. Note that this time the complete system is linearized and therefore the coupling signals no longer exist.

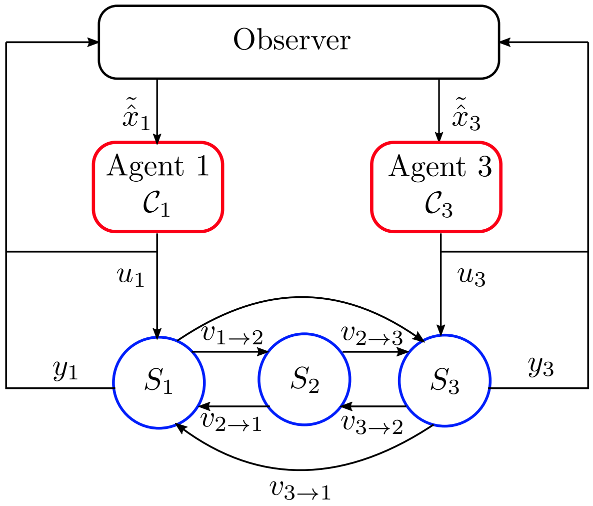


Figure 2.5: Synoptic view of a observer-based decentralized control scheme. In this scenario, only S_1 and S_3 receive the state estimates to compute their control input in order to regulate the outputs.

Assuming that the pair (A, C) is observable, the Luenberger observation gain L can be computed by using the Matlab subroutine ***dlqr()***. Consequently, the observer's equation is formulated as follows:

$$\tilde{\hat{x}}^{+} = (A - L \cdot C) \cdot \tilde{\hat{x}} + B \cdot \tilde{u} + L \cdot \tilde{y} \tag{2.31}$$

The estimated state $\tilde{\hat{x}} \in \mathbb{R}^{n_x}$ is updated by (2.31). Note that $\hat{x}$ is nothing but the concatenation of all the individual local state $\tilde{\hat{x}}_s$ (for $s \in \mathcal{N}$). Therefore, the observer can allocate these local estimates $\hat{x}_s$ to the corresponding subsystem S_s. Fig. 2.5 illustrates an observer-based decentralized control scheme where the centralized observer is synthesized and implemented in order to estimate the states of the local subsystems.

If ever an agent needs to implement a nonlinear controller synthesized from a model whose states are not identical to those of the linear model, the corresponding extended Kalman observer (also called extended Kalman filter (EKF)) is systematically added to estimate the states of this nonlinear model. The addition of the EKF is due to the fact that the linear model resulting from the use of the Simcryogenics library usually has a high number of states, which is not practical to have the nonlinear model with the same number of states. Fig. 2.6 illustrate the scenario where a nonlinear controller and a corresponding EKF are integrated in the control scheme. Note that the EKF need to receive the estimate of incoming coupling signal $\hat{v}_3^{in}$ in order to estimate the state $\hat{x}_3$.

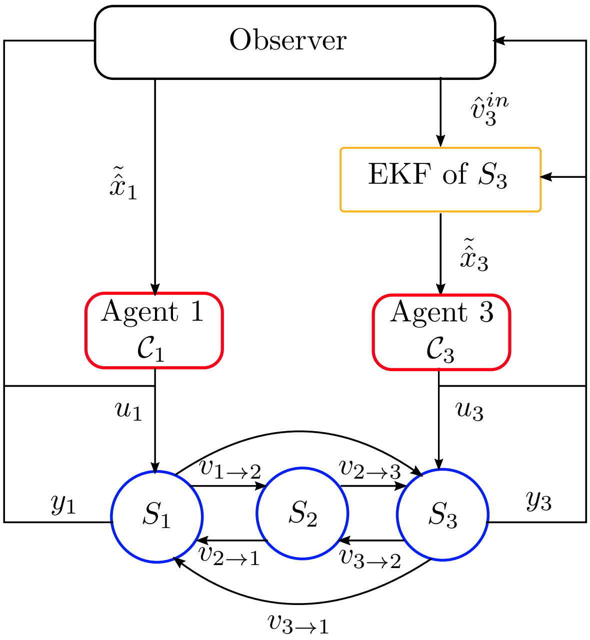


Figure 2.6: Synoptic view of a observer-based decentralized control scheme with nonlinear observer. In this scenario, S_3 implements a nonlinear observer EKF.

CHAPTER 3

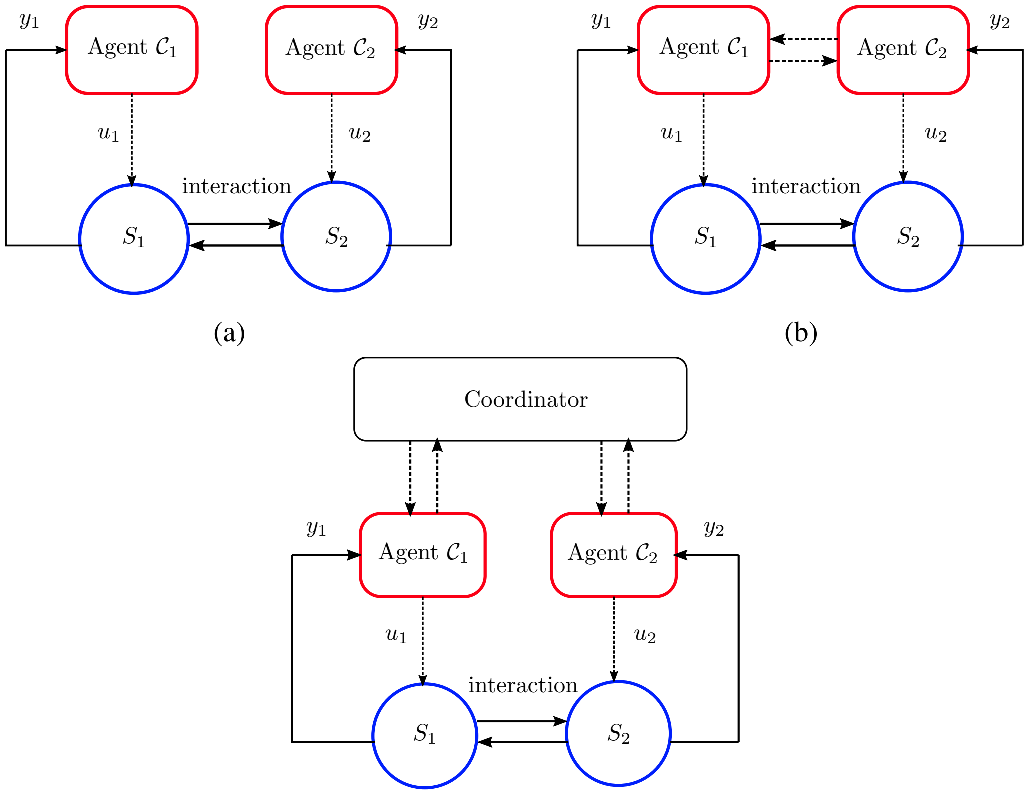
Fixed-point iteration based hierarchical control

Abstract This chapter presents an extension of a recently proposed hierarchical control framework applied to control a cryogenic refrigerator. The original work has validated this framework in the case where each subsystem is regulated by a local agent that implements unconstrained linear controllers. Hence, The extension concerns the validation of the framework in the presence of both nonlinear models and controls. It is also shown that real-time handling of these features requires a specific complexity reduction technique. This technique aims to perform the distribution of the optimization process over cyclically changed decision variables aiming at limiting the number of iterations per updating period. Numerical simulations are proposed in order to show the impact of the parameter choices and to assess the real-time implementability of the proposed framework. Moreover, numerical simulations also demonstrate that the overall system is well coordinated by this approach in the case where some of the subsystems are not regulated.

3.1 State of art

In order to address the problems that have been mentioned in Chapter 1, many works have studied a considerable number of non-centralized control architectures [33, 34], from which three typical structures can be derived:

- **Decentralized control structure (Fig: 3.1a):** The subsystems in this structure are individually controlled by the agents. Since there is no communication between these agents (or regulators), each of them attempts to optimize its local problem without taking into account to the coupling effects they have on each other.
- **Distributed control structure (Fig: 3.1b):** In this kind of structure, the agents communicate with each other to find an optimal solution. The information transmitted between subsystems can be either the predicted evolution of system states ($\boldsymbol{x}_1$ and $\boldsymbol{x}_2$) or the predicted control actions ($\boldsymbol{u}_1$ and $\boldsymbol{u}_2$). In the former, the local agents only need to know the subsystem dynamic under their direct control, while in the latter, the dynamic models of the subsystems that influence their decisions are needed.



(a) (b)

(c)

Figure 3.1: Synoptic view of the non-centralized control architectures: decentralized (a), distributed (b), hierarchical control structure (c).

- **Hierarchical control structure (Fig: 3.1c):** This approach is characterized by a two-level structure. At the local level, the subsystems are linked by the outputs (coupling signals) of each other and controlled by their corresponding agents according to the received set-point. At the coordination level, the coordinator exchanges information with these agents in order to optimize global performance while ensuring that the coherence constraints on the coupling effects are respected.

In general, distributed and hierarchical control approaches can address some problems mentioned in Chapter 1. Although the distributed one is mainly applied in the large-scale system control domain because of its simplicity [35, 36, 37], it is often implemented for applications where the subsystems are weakly coupled, which is not valid for cryogenic systems. Besides, if fully distributed control (connections between all subsystems) is considered, the implied network traffic will become complicated when there are too many subsystems communicating with each other, which makes the implementation of this method much more challenging.

On the contrary, the communication traffic issue implied by the hierarchical framework is lightened because the information is only exchanged between the coordinator and the local layer. The well-known communication protocol between the two layers is the iterative "price coordination" method [38, 39]. In this method, the coordinator sets the prices as the Lagrange multipliers of the coherence constraints in the global optimization problem imposed by the information (states, inputs, and outputs profiles) sent by the local agents. Upon receiving these optimal prices, the local agents recompute the control profile and the corresponding profiles of states and outputs. The iterations are stopped when the coherence constraint on the coupling signals is satisfied. However, this method encounters some disadvantages, as shown below:

- In most works, it is assumed that the coordination layer is built based on the MPC decentralized scheme, which is not generally true in real-life applications. Indeed, in cryoplants, most of the subsystems are controlled by classical controllers such as PID or LQR.
- As long as MPCs are concerned in the local layer, the coordination layer is added to retrieve the equivalent centralized optimal solution by incorporating the interaction effects in the local optimization problems. Their control problems are generally synthesized with a fixed set of weighting matrices. These parameters are rarely modified since it will induce a complete change in the overall framework. The defined optimization problem might not be compatible with the system operation, especially when the controlled system has several modes that require different weighting matrices. Hence, the need for a framework that allows to switch flexibly between the different modes is an emerging topic.

In order to address these drawbacks, a method employing the hierarchical structure, which has been developed by [1], is briefly described below:

The targeted hierarchical control scheme is constructed by basing on the existing decentralized control layer, in which the local controllers receive appropriate set-points r_s from the coordinator that minimizes a central cost J_c. Furthermore, the proposed framework satisfies the **hierarchical requirements** that are defined below:

- The agents in the network communicate exclusively with the coordinator.
- The coordinator does not know any information concerning the mathematical models of the subsystems and the details of their controllers, which is exactly the modular privacy-preserving requirement.
- The different ***operation modes*** (which will be clarified in the next section) are handled by adjusting some coefficients of the central cost disposed at the coordination layer without changing the local controllers.

Although the method proposed by [1] has been successfully tested for a cryogenic application, there are still some challenges that need to be addressed, which are given below:

1. The original work is only proved to be valid where the local agents implement unconstrained linear MPCs. However, there are cases where the non-linearities of the subsystems and the actuator constraints are required to be incorporated into the control problem in order to ensure control performance. Adding these two terms (actuator constraints and non-linearities) increases significantly the computation burden that could make the implementation unfeasible. To address this issue, this chapter and the next chapter will propose some techniques that are to be considered when this type of issues arises.

2. In addition, the inner-loop, which is the communication between the coordinator and the local agents, is only proved to converge when unconstrained linear controllers are involved in the local layer. This convergence is achieved thanks to an innovative filter synthesized by using the mathematical information of the local agents, which violates the modular privacy-preserving requirement. Also, the communication can diverge if the filter is not compatible with the constrained nonlinear versions of the local controllers. Hence, a model-free method, which colorgreenenhances the convergence of the inner loop, will be introduced in Chapter 5.

The next section will recall the fixed-point iteration based control method introduced in [1], which is the key topic of this book.

3.2 Fixed-point-iteration based hierarchical control formulation

Let's take an example where the proposed hierarchical control framework is applied to a network of three interacting subsystems $S_{s\in\mathcal{N}}$ illustrated by Fig. 3.2. The definition of the controlled / uncontrolled subsystem subset $\mathcal{N}^{ctr}$ / $\mathcal{N}^{unc}$, the neighbor subset $\mathcal{N}_s$, and the coupling signals $v_{s\rightarrow s'}$ are also reused in this scenario.

With a slight abuse of notation, x_s, u_s, $v_{s\rightarrow s'}$, and y_s are used to express respectively the deviations from the operating points x_s^{op}, u_s^{op}, $v_{s\rightarrow s'}^{op}$ and y_s^{op}. In addition, the operation described in this section is processed during an updating period $[k, k+1]$, the initial state $x_s(k)$ is thus frozen and dropped in all equations that are concerned.

Let $\boldsymbol{v}_s^{in}$ and $\boldsymbol{v}_s^{out}$ represent the incoming and outgoing coupling profile into and from the subsystem S_s, respectively. Their equations are given below:

$$\boldsymbol{v}_s^{in} := \underset{s'\in\mathcal{N}_s}{\text{col}}\,(\boldsymbol{v}_{s'\rightarrow s}); \quad \boldsymbol{v}_s^{out} := \underset{s'|s\in\mathcal{N}_{s'}}{\text{col}}\,(\boldsymbol{v}_{s\rightarrow s'}) \tag{3.1}$$

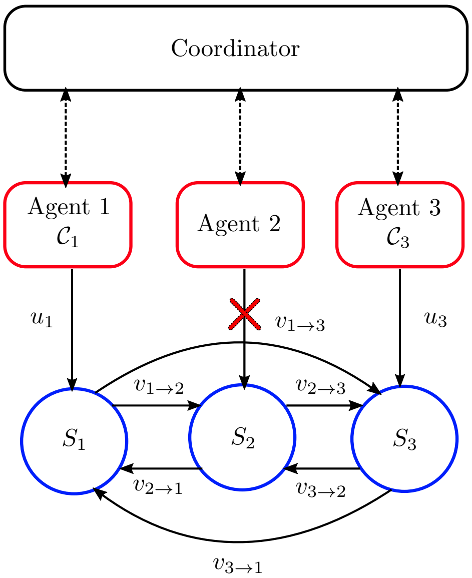


Figure 3.2: Hierarchical structure applied to a network of three interconnecting subsystems. The defined sets correspond to this example are: $\mathcal{N} = \{1, 2, 3\}$, $\mathcal{N}^{ctr} = \{1, 3\}$, $\mathcal{N}^{unc} = \{2\}$, $\mathcal{N}_1 = \{2, 3\}$, $\mathcal{N}_2 = \{1, 3\}$ and $\mathcal{N}_3 = \{1, 2\}$. The double-arrow dash lines represent the communication between the coordinator and the agents.

In this hierarchical framework, some assumptions regarding the process occurring at the local layer need to be introduced:

Assumption

Each subsystem S_s, when given

- a presumed incoming profile $\boldsymbol{v}_s^{in}$ and
- a given individual set-point r_s (required if $s \in \mathcal{N}^{ctr}$),

can compute what would be:

- Its control profile $\boldsymbol{u}_s$ (if it has),
- Its resulting outgoing profile $\boldsymbol{v}_s^{out}$,
- Its contribution J_s to the central cost.

The central cost is assumed to be of the form:

$$J_c(r, \boldsymbol{v}^{in}) := \sum_{s \in \mathcal{N}} J_s(r, \boldsymbol{v}_s^{in}) \tag{3.2}$$

where $r := \text{col}_{s\in\mathcal{N}^{ctr}}(r_s)$ and $\boldsymbol{v}^{in} := \text{col}_{s\in\mathcal{N}}(\boldsymbol{v}_s^{in})$. From the coordinator's point of view, the local costs contributed by the uncontrolled subsystems to the coordinator should be considered as the functions of the given set-point r and the local incoming coupling profile $\boldsymbol{v}_s^{in}$.

The local agents that have outputs to be regulated implement the controllers denoted by C_s, for $s \in \mathcal{N}^{ctr}$. For any set-points r_s and incoming coupling profiles $\boldsymbol{v}_s^{in}$ sent by the coordinator, the local agents can compute the corresponding control profile, namely:

$$\boldsymbol{u}_s = C_s(r_s, \boldsymbol{v}_s^{in}), \quad \forall s \in \mathcal{N}^{ctr} \tag{3.3}$$

It should be recalled that the subsystem's state x_s is omitted in (3.3). The global outgoing coupling profiles are also dependent on the given set-point $r := \text{col}_{s\in\mathcal{N}}(r_s)$, as shown below:

$$\boldsymbol{v}^{out} := \underset{s\in\mathcal{N}}{\text{col}}(\boldsymbol{v}_s^{out}), \quad \text{with} \quad \boldsymbol{v}_s^{out} := \boldsymbol{g}_s^{out}(\boldsymbol{u}_s, \boldsymbol{v}_s^{in}) \tag{3.4}$$

where $\boldsymbol{g}_s^{out}(\cdot)$ ($\forall s \in \mathcal{N}$) are the local functions that compute the local outgoing coupling profile $\boldsymbol{v}_s^{out}$. Note that for $s \in \mathcal{N}^{unc}$, the control profiles $\boldsymbol{u}_s$ appearing in (3.4) are simply considered to have null size, i.e, $\boldsymbol{u}_s = \emptyset$.

Similarly, let $\boldsymbol{v}^{in}$ denote the vector that gathers the incoming coupling signals of all the subsystems, namely:

$$\boldsymbol{v}^{in} = \underset{s\in\mathcal{N}}{\text{col}}(\boldsymbol{v}_s^{in}) \tag{3.5}$$

It should be noted that the elements of the outgoing coupling profile $\boldsymbol{v}^{out}$ are also those of the incoming coupling profile $\boldsymbol{v}^{in}$ but arranged in a different order. Indeed, both $\boldsymbol{v}^{in}$ and $\boldsymbol{v}^{out}$ are composed of all the profiles of the form $\boldsymbol{v}_{s\rightarrow s'}$. There is a permutation matrix G_{in} such that:

$$\boldsymbol{v}^{in} := G_{in} \cdot \boldsymbol{v}^{out} \tag{3.6}$$

Then, injecting (3.4) into (3.6) yields the following so-called coherence constraint:

$$\boldsymbol{v}^{in} = G_{in} \cdot \underset{s\in\mathcal{N}}{\text{col}}(\boldsymbol{v}_s^{out}) = G_{in} \cdot \underset{s\in\mathcal{N}}{\text{col}}(\boldsymbol{g}_s^{out}(\boldsymbol{u}_s, \boldsymbol{v}_s^{in})) = \boldsymbol{g}^{in}(r, \boldsymbol{v}^{in}) \tag{3.7}$$

which governs the interaction dynamic of the overall system specified to the given set-point r.

As long as the coherence constraint is satisfied, the output profiles associated to the given set-point r can be computed, namely:

$$\boldsymbol{y}_s = \mathbf{h}_s(\boldsymbol{u}_s, \boldsymbol{v}_s^{in}, \boldsymbol{w}_s) \tag{3.8}$$

At this stage, the central problem at the coordination level can be defined:

$$r^{\text{opt}} = \underset{r\in\mathcal{R}}{\text{argmin}} \; J_c(r, \boldsymbol{v}^{in}) \tag{3.9}$$

$$\text{subject to: } \boldsymbol{v}^{in} = \boldsymbol{g}^{in}(r, \boldsymbol{v}^{in}) \tag{3.10}$$

where $\mathcal{R}$ is the admissible set of r, namely:

$$\mathcal{R} = \{r \quad | \quad r_{\min} \le r \le r_{\max}\} \tag{3.11}$$

with $r_{\min}, r_{\max} \in \mathbb{R}^{n_r}$ being the priori defined bounds on possible values of the set-points.

The central cost $J_c(r, \boldsymbol{v}^{in})$ is the sum of the local contributions J_s of the subsystems, namely:

$$J_c(r, \boldsymbol{v}^{in}) := \sum_{s \in \mathcal{N}} J_s(r, \boldsymbol{v}_s^{in}) \tag{3.12}$$

The local contribution cost can either express the economic cost or the constraint violation cost as follows:

- For economic cost:

  $$J_s(r, \boldsymbol{v}_s^{in} | r_s^d) = \|\boldsymbol{y}_s - r_s^d\|^2_{Q_c^{(s)}} + \|\boldsymbol{u}_s\|^2_{R_c^{(s)}} \tag{3.13}$$

  where $r_s^d \in \mathbb{R}^{n_{ys}}$ is the desired set-point of the subsystem S_s.

- For constraint violation cost

  $$J_s(r, \boldsymbol{v}_s^{in} | \overline{\boldsymbol{y}}_s, \underline{\boldsymbol{y}}_s) = \|\max(\boldsymbol{y}_s - \overline{\boldsymbol{y}}_s, 0)\|^2_{Q_{cstr}^{(s)}} + \|\max(\underline{\boldsymbol{y}}_s - \boldsymbol{y}_s, 0)\|^2_{Q_{cstr}^{(s)}} \tag{3.14}$$

  where $\underline{\boldsymbol{y}}_s \in \mathbb{R}^{n_{ys}}$/ $\overline{\boldsymbol{y}}_s \in \mathbb{R}^{n_{ys}}$ is the minimum value/maximum value on the output $\boldsymbol{y}_s$ of the subsystem S_s.

The ***operation modes*** mentioned in the previous section can be taken into account by using different set-points and different weighting matrices in the centralized cost. The role of the coordinator is to control the system through two operation modes which are described hereafer:

1. In the first mode, the objective is to regulate the system around the nominal point $x = 0$ in spite of the unmeasured disturbances (disturbance-rejection mode). This is the main objective of the cryogenic refrigerator.
2. In the second mode, the coordinator can temporarily drive the system to a different steady-state corresponding to a new set-point $\tilde{y} \neq 0$. For instance, the operator might decide to change the liquid helium level in the bath or some temperatures at some specific locations.

! Attention

It is essential to note that the defined central cost (3.12) is a combination of the elementary local costs J_s representing the economic and security indicators related to a given set of setpoints r. In fact, the desired setpoint given by the operator may not be the optimal one considering these two aspects. Hence, the key idea is that the coordinator solves the optimization problem (3.9)-(3.10) for the optimal set-point r^{opt}, while guarantying the defined **hierarchical requirement**.

Based on that, to solve (3.9)-(3.10), the coordinator can implement an algorithm that is separated into two procedures, namely:

Overview of the fixed-point based hierarchical control method

Estimating central cost $J_c(r, \boldsymbol{v}^{in})$: For any set-points $r_{s\in\mathcal{N}^{ctr}}$ sent by the coordinator to the agents, there is a communication process between the coordinator and the agents in order to find the central cost $J_c(r, \boldsymbol{v}^{in})$ consistent with the coherence constraint.

Optimizing the central cost: Having the process to estimate the central cost J_c for any set-point sent by the coordinator, any derivative-free optimization algorithm can be used to find the optimal set-point r^{opt}, such as BOBYQA [40], genetic algorithm [41], etc. This chapter proposes a simple but efficient solver for solving optimization problems based on quadratic approximation.

The next sections will describe the mentioned procedures.

3.3 Estimating the central cost

3.3.1 Fixed-point iteration based communication:

At this time, we have seen that for a given set-point r, there exists the corresponding local cost by which the agents contribute to the central cost, namely:

$$J_c(r, \boldsymbol{v}^{in}) = \sum_{s\in\mathcal{N}} J_s(r, \boldsymbol{v}_s^{in}) \quad \text{subject to } \boldsymbol{v}^{in} = \boldsymbol{g}_{out}(r, \boldsymbol{v}^{in}) \tag{3.15}$$

These local contributions J_s depend on the output and input profiles of the subsystems, which can be easily computed if the agents have the exact relevant incoming coupling profiles $\boldsymbol{v}_s^{in}$ that satisfy the coherence constraint (3.7). Moreover, this condition clearly exhibits a fixed-point like function $p = G(p)$ that can be solved by fixed-point iteration for the exact $\boldsymbol{v}^{in}$ associated to r.

To do so, the coordinator can begin with some initial guesses:

$$\boldsymbol{v}_s^{in,(\sigma=0)}, \quad \forall\, s \in \mathcal{N} \tag{3.16}$$

that are sent to the agents. Then, they can compute the corresponding estimates of out-coming coupling profiles that will be sent back to the coordinator.

$$\hat{\boldsymbol{v}}_s^{out,(\sigma)} = \boldsymbol{g}_s^{out}(C_s(r_s, \boldsymbol{v}_s^{in,(\sigma)}), \boldsymbol{v}_s^{in,(\sigma)}) \qquad \forall\, s \in \mathcal{N}^{ctr} \tag{3.17}$$

$$\hat{\boldsymbol{v}}_s^{out,(\sigma)} = \boldsymbol{g}_s^{out}(\varnothing, \boldsymbol{v}_s^{in,(\sigma)}) \qquad \forall\, s \in \mathcal{N}^{unc} \tag{3.18}$$

After receiving these new estimates, the coordinator gathers all the individual vector $\hat{\boldsymbol{v}}_s^{out,(\sigma)}$ into $\hat{\boldsymbol{v}}^{out(\sigma)}$. Note that the coordination can reconstruct the estimate of the incoming coupling profile $\hat{\boldsymbol{v}}^{in}$ by using the permutation matrix G_{in} in (3.6), that is:

$$\hat{\boldsymbol{v}}^{in,(\sigma)} = G_{in} \cdot \hat{\boldsymbol{v}}^{out,(\sigma)} \tag{3.19}$$

Then, this version is updated it by using a filter, namely:

$$\boldsymbol{v}^{in,(\sigma+1)} = (\mathbb{I} - \Pi) \cdot \boldsymbol{v}^{in,(\sigma)} + \Pi \cdot \hat{\boldsymbol{v}}_s^{in,(\sigma)} \tag{3.20}$$

where Π is the filter matrix that enhances the convergence of the iteration. The synthesis of this matrix will be discussed in the next subsection. As a matter of fact, the coordinator can split the updated version of the outgoing coupling profile $\boldsymbol{v}^{in,(\sigma+1)}$ into the individual incoming coupling profiles $\boldsymbol{v}_s^{in,(\sigma+1)}$ that will be sent to the local agents.

Upon receiving the new incoming coupling profile, the local agents can repeat the described procedure until the iteration converges toward some fixed-point $\boldsymbol{v}^{in,(\infty)}$, equivalently:

$$\lim_{\sigma \to \infty} \boldsymbol{v}^{in,(\sigma)} = \boldsymbol{v}^{in,(\infty)} \tag{3.21}$$

In practice, the iteration can be stopped if the termination criteria $|\boldsymbol{v}^{in,(\sigma+1)} - \boldsymbol{v}^{in,(\sigma)}| < \epsilon_{\max}$ or $\sigma \le \sigma_{\max}$ are satisfied. Upon the convergence, the agents can compute its local costs $J_{s \in \mathcal{N}}$ and send them to the coordinator, by which the coordinator computes the central cost $J_c(r, \boldsymbol{v}^{in,(\infty)})$.

3.3.2 Designing the filter matrix Π

The previous section has described the fixed-point based-iteration protocol whose convergence is enhanced by using the filter matrix Π. This section will describe the method to synthesize the mentioned filter.

Assume that all subsystem models are linear time-invariant and that subsystems having control inputs implement classical linear control laws such as PID-based control design, LQR, or unconstrained linear MPC type. Therefore, the control profiles $\boldsymbol{u}_s$ ($\forall s \in \mathcal{N}^{ctr}$) are expressed as linear (disturbance-free) equations presented below, given the current guess of the incoming coupling signal profiles $\boldsymbol{v}_s^{in}$ at the fixed-point iteration number σ:

$$\boldsymbol{u}_s^{(\sigma)} := K_s^{(x)} \cdot x_s(k) + K_s^{(r)} \cdot r_s + K_s^{(v)} \cdot \boldsymbol{v}_s^{in,(\sigma)} \tag{3.22}$$

On the other hand, the estimate of the outgoing coupling profiles can be derived from the linear dynamic equations by using the above control profiles (3.22):

$$\hat{\boldsymbol{v}}_s^{out,(\sigma)} := \Phi_s^{(x)} \cdot x_s(k) + \Phi_s^{(u)} \cdot \boldsymbol{u}_s + \Phi_s^{(v)} \cdot \boldsymbol{v}_s^{in,(\sigma)} \tag{3.23}$$

with $s \in \mathcal{N}^{ctr}$. Note that for the subsystems S_s with $s \in \mathcal{N}^{unc}$, the term $\Phi_s^{(u)} \cdot \boldsymbol{u}_s$ does not exist.

Combining (3.23) and (3.22), the following equation of the coupling profile is obtained:

$$\hat{\boldsymbol{v}}_s^{out,(\sigma)} := \Psi_s^{(x)} \cdot x_s(k) + \Psi_s^{(v)} \cdot \boldsymbol{v}_s^{in,(\sigma)} + \Psi_s^{(r)} \cdot r_s \tag{3.24}$$

By following similarly the steps that has been explained in section 3.2, the estimate of the global incoming coupling profile can be expressed by the following equation:

$$\hat{\boldsymbol{v}}^{in,(\sigma+1)} = \overline{M}^{(v)} \cdot \boldsymbol{v}^{in,(\sigma)} + \overline{M}^{(x)} \cdot x(k) + \overline{M}^{(r)} \cdot r \tag{3.25}$$

with $\hat{\boldsymbol{v}}^{in} = \text{col}_{s\in\mathcal{N}}(\hat{\boldsymbol{v}}_s^{in})$ and $x = \text{col}_{s\in\mathcal{N}}(x_s)$. The matrices $\overline{M}^{(v)}$, $\overline{M}^{(x)}$ and $\overline{M}^{(r)}$ are coming from the matrices of the linear models of the subsystems. Equation (3.25) could be seen as dynamical equation of the fixed-point iteration and it does not necessarily converge.

In order to enforce the convergence of the fixed-point iteration, [1] proposed an advanced filter (which is also called mixing method [42, 43]) to update the incoming coupling profile, namely:

$$\boldsymbol{v}^{in,(\sigma+1)} = (\mathbb{I} - \Pi) \cdot \boldsymbol{v}^{in,(\sigma)} + \Pi \cdot \hat{\boldsymbol{v}}^{in,(\sigma+1)} \tag{3.26}$$

The convergence condition for a choice of Π will be determined thereafter. By injecting (3.25) in (3.26), we obtain:

$$\boldsymbol{v}^{in,(\sigma+1)} = \left[\mathbb{I} - \Pi \cdot (\mathbb{I} - \overline{M}^{(v)})\right] \boldsymbol{v}^{in,(\sigma)} + \Pi \cdot \left[\overline{M}^{(x)} \cdot x(k) + \overline{M}^{(r)} \cdot r\right] \tag{3.27}$$

This clearly shows that the convergence of the fixed-point iteration is conditioned by the spectrum radius of the matrix $\left[\mathbb{I} - \Pi \cdot (\mathbb{I} - \overline{M}^{(v)})\right]$. More precisely, the fixed-point iteration converges if and only if:

$$\rho\left(\left[\mathbb{I} - \Pi \cdot (\mathbb{I} - \overline{M}^{(v)})\right]\right) < 1 \tag{3.28}$$

where $\rho(Z)$ denotes the spectrum radius of the matrix Z, namely:

$$\rho(Z) := \max_i |\lambda_i(Z)| \tag{3.29}$$

with λ being the eigenvalues of matrix Z.

The condition (3.28) can be satisfied if the pair $\left(\mathbb{I}, [\mathbb{I} - \overline{M}^{(v)}]^T\right)$ is controllable. If this is the case, the appropriate matrix Π can be obtained by using the discrete linear quadratic design tools (such as the subroutine MATLAB's **dlqr** utility). The whole process that estimates the central cost associated to a given set-point by using the mixing method is described in Algorithm 1.

Algorithm 1 Mixing method for fixed-point iteration for evaluating the central cost associated to a given set-point $r_{s \in \mathcal{N}^{ctr}}$

```
Initialize:
    $r = \mathrm{col}_{s \in \mathcal{N}^{ctr}}\, r_s$, $\boldsymbol{v}_s^{in,(0)} \leftarrow 0$, $s = 1, \ldots, n$, $\sigma \leftarrow 0$; $\epsilon \leftarrow \infty$;
Coordinator sends $r_s$ and $\boldsymbol{v}_s^{in,(0)}$ to the subsystems;
while ($\sigma \le \sigma_{\max}$) and ($\epsilon \le \epsilon_{\max}$) do
    for $s \in \mathcal{N}$ do                    ▷ Parallel operation performed by the subsystems
        Subsystem $S_s$ computes $\hat{\boldsymbol{v}}_s^{out,(\sigma)}$ and sends to coordinator;
    end for
    Coordinator concatenates $\hat{\boldsymbol{v}}_s^{out,(\sigma)}$ into $\hat{\boldsymbol{v}}^{out,(\sigma)}$;
    Coordinator computes $\hat{\boldsymbol{v}}^{in,(\sigma+1)} := G_{in} \cdot \hat{\boldsymbol{v}}^{out,(\sigma)}$;
    Coordinator computes the filtered version $\boldsymbol{v}^{in,(\sigma+1)}$ by (3.26);
    Coordinator distributes $\boldsymbol{v}_s^{in,(\sigma+1)}$ to the subsystems $S_s$, for $s \in \mathcal{N}$;
    $\sigma \leftarrow \sigma + 1$;
    $\epsilon \leftarrow \max(|\hat{\boldsymbol{v}}^{in,(\sigma+1)} - \hat{\boldsymbol{v}}^{in,(\sigma)}|)$;
end while
for $s \in \mathcal{N}$ do
    Subsystem $S_s$ computes $\hat{\boldsymbol{y}}_s$ then $J_s$ and sends $J_s$ to the coordinator;
end for
Coordinator computes $J_c(r, \boldsymbol{v}^{in,(\sigma)})$ by (3.12);
```

Tips

If the considered subsystems employ the nonlinear models, the filter matrix Π can be computed from their linearized model around the operating points (x_s^{op}, y_s^{op}). Indeed, the convergence is only proved if their equations can be expressed as such linear functions. However, it is essential to note that the stability of the iterations holds locally, and the efficiency of this method in cases other than linear settings can be verified a posteriori.

! Attention

The synthesis of the filter matrix Π violates the modular privacy-preserving requirement. Clearly, it uses the information of the underlying dynamics that are condensed in the definition of the matrices $\overline{M}^{(x)}$ and $\overline{M}^{(v)}$ invoked in (3.25).

3.4 Optimizing the central cost

The previous section has described the fixed-point-iterations based method that allows the coordinator to compute the central cost for a **given set-point** r. Recall, however, that the role of the coordinator is to optimize the choice of the auxiliary set-points so that the central cost can be minimized. This can be done by using any derivative-free optimization algorithm such as BOBYQA [40], genetic algorithm [41], etc. However, using these algorithms can increase the computational burden that makes the whole algorithm unfeasible in real-time implementations. Instead, this section explains how the coordinator can use successive evaluations of the central cost for different candidate auxiliary set-points to construct a quadratic approximation of the central cost (as a function of the auxiliary set-points at the current sampling time k) in order to derive a candidate optimal auxiliary set-point r^{opt}.

3.4.1 Approximating the central cost

Using the fixed-point iteration, the coordinator can compute for each auxiliary set-point r the corresponding value of the central cost:

$$J_c(r) \tag{3.30}$$

after convergence of the fixed-point iteration.

The central problem in the coordination layer can now be recalled:

$$r^{\text{opt}} = \underset{r\in\mathcal{R}}{\operatorname{argmin}}\, J_c(r) \tag{3.31}$$

In order to solve (3.31), the central cost (3.30) will be approximated by a quadratic function, namely:

$$\hat{J}_c = \frac{1}{2} r^T Q r + f^T r + c \tag{3.32}$$

where $Q \in \mathbb{R}^{n_r \times n_r}$, $f \in \mathbb{R}^{n_r}$ and $c \in \mathbb{R}$, with n_r being the dimension of vector r.

These unknown parameters can be identified if the coordinator disposes of the values of the central cost at, at least $(n_r+1)(n_r+2)/2$ different auxiliary set-points. The remaining part of this section is devoted to explaining the way this is done by the coordinator. Note that this is a single possibility among many other possibilities of optimizing a black-box given function through different evaluations of its values at a set of possible points within its domain of definition. This is linked to the general domain of derivative-free optimization.

The approximation can be achieved by evaluating the central cost at every candidate set-point r in a moving grid denoted by $\mathbf{G}(k)$. More precisely, this grid is constructed around the suboptimal solution found at the last instant $r^{\text{opt}}(k-1)$ and is bounded by a so-called trust-region size $\rho(k)$. Depending on the relevance of quadratic approximation, the size ρ of the trust-region is modified, which will be described in section 3.4.2. At each sampling instant k, the grid $\mathbf{G}(k)$ of auxiliary set-points for the evaluation of the central cost is defined **around the previous optimal value $r^{\mathbf{opt}}(k-1)$** as follows:

$$\mathbf{G}(k) := \text{Pr}\left(r^{\text{opt}}(k-1) + \Delta(\rho(k-1)), \mathcal{R}\right) \tag{3.33}$$

where

- $\mathcal{R}$ is an admissible set of r, namely:

  $$\mathcal{R} = \{r \quad | \quad r_{\text{min}} \leq r \leq r_{\text{max}}\} \tag{3.34}$$

  where $r_{\text{min}}, r_{\text{max}} \in \mathbb{R}^{n_r}$ are a priori defined bounds on possible values of the set-points.
- for a discrete subset $\mathbb{D} \subset \mathbb{R}^{n_r}$, the notation $\text{Pr}\,(\mathbb{D}, \mathcal{R})$ denotes the discrete set obtained by projecting all the elements of $\mathbb{D}$ on the hypercube $\mathcal{R}$.
- $\rho \in \mathbb{R}_+$ is strictly positive real (size of the trust region where the quadratic approximation is presumably relevant).
- $\Delta(\rho)$ is a discrete set of displacements in $\mathbb{R}^{n_r}$ defined around 0 with distances that are proportional to ρ so that $r^{\text{opt}}(k-1) + \Delta(\rho)$ represent the set of different auxiliary set-points around the previous optimal value to be visited and where the cost is to be evaluated). More precisely, the subset $\Delta(\rho) \subset \mathbb{R}^{n_r}$ is defined by:

  $$\Delta(\rho) := \left\{-\frac{(m-1)}{2}\rho, \dots, -\rho, 0, \rho, \dots, \frac{(m-1)}{2}\rho\right\}^{n_r} \tag{3.35}$$

where m is supposed to be odd so that $\frac{(m-1)}{2}$ is a natural number. The trust region size ρ is updated at each instant k, which will be described later. Recall that the identifiability of the quadratic form coefficients is possible provided that $m^{n_r} \geq (n_r+1)(n_r+2)/2$.

Based on the above definitions, the grid $\mathbf{G}(k)$ is constructed by using (3.33), the evaluation of the central cost $J_c(\cdot)$ at every set-point $r \in \mathbf{G}(k)$ is performed by using the fixed-point methodology introduced in section 3.3.2. Note that the number of set-points to be evaluated n_{ev} can be chosen to be equal to the sufficient number $(n_r+1)(n_r+2)/2$. The values

$$J_c(r^{(j)}), \quad j = 1, \dots, n_{ev} \leq m^{n_r}$$

enable to compute the parameters of the quadratic form:

$$\text{(Coordinator)} \min_{Q,f,c} \sum_{j=1}^{n_{ev}} \left| J_c(r^{(j)}) - \left[\frac{1}{2} \|r^{(j)}\|_Q^2 + f^T r^{(j)} + c \right] \right| \tag{3.36}$$

Once Q, f and c are available, a candidate optimal set-point r_c^{opt} (that minimizes the quadratic approximation) can be computed. Note, however, that since the central cost is not necessarily quadratic, this candidate optimal cost does not necessarily induce a decrease in the central cost. This can happen when the trust-region parameter ρ is too large for the quadratic approximation to be relevant. In such case, the size ρ should be reduced. This mechanism is discussed in the next section.

3.4.2 Trust region updating law of ρ

As mentioned previously, the parameter ρ defines the size of the neighborhood of the current desired set-point r^d over which the better value is computed based on the current quadratic approximation of the cost function. On one hand, ρ must be sufficiently high to ensure a rapid decrease of the cost value. On the other hand, small values of ρ might be required in order for the quadratic approximation to be relevant. Hence, ρ should be updated accordingly: ρ is increased if the quadratic approximation induces a decrease of the cost function while ρ is decreased otherwise.

Concretely, the following quadratic problem is first solved to obtain the candidate value $r_c(k)$

$$r_c^{\text{opt}}(k) = \operatorname*{argmin}_{r \in \mathscr{P}(k)} \hat{J}_c(r) \tag{3.37}$$

where $\mathscr{P}(k)$ is given by:

$$\mathscr{P}(k) := \text{Conv}\{\Pr\left(r_d(k) + \Delta(\rho(k-1)), \mathcal{R}\right)\} \tag{3.38}$$

Once the candidate $r_c^{\text{opt}}(k)$ is obtained, the corresponding cost is computed by executing the algorithm 1 to obtain $J_c(r_c^{\text{opt}})$. The quadratic approximation is said to be relevant if it meets the condition below:

$$J_c(r_c^{\text{opt}}) < \min\{J_c(r^{(j)}) \mid r^{(j)} \in \mathbf{G}(k)\} \tag{3.39}$$

Therefore, the trust-region size ρ is updated according to:

$$\rho(k) := \begin{cases} \beta^+ \cdot \rho(k-1) & \text{if (3.39) is satisfied} \\ \beta^- \cdot \rho(k-1) & \text{otherwise} \end{cases} \tag{3.40}$$

where $\beta^+ \geq 1$ and $\beta^- \in (0, 1)$ denote respectively the expansion and the contraction factors. Finally, the updating law for r^{opt} is given by:

$$r^{\text{opt}}(k) := \begin{cases} r_c^{\text{opt}}(k) & \text{if (3.39) is satisfied} \\ r^{\text{opt}}(k-1) & \text{otherwise} \end{cases} \tag{3.41}$$

where $r^{\text{opt}}(k-1)$ is the solution found at the previous instant $k-1$. The so adopted set-point $r^{\text{opt}}(k)$ is then sent to the subsystems with an *end-of-iterations* flag, which allows the agents to compute their corresponding control profiles. Finally, the first action in each control profile, namely:

$$u_s(k) := [\mathbb{I}_{n_s^u}, \mathbb{O}_{n_s^u}, \dots, \mathbb{O}_{n_s^u}]\boldsymbol{u}_s^{\text{opt}}(k) \tag{3.42}$$

with $\boldsymbol{u}_s^{\text{opt}}(k) = C_s(r_s^{\text{opt}}(k)|\boldsymbol{v}_s^{in,(\infty)})$ $(\forall s \in \mathcal{N}^{ctr})$, is applied to subsystem S_s during the sampling period $[k, k+1]$.

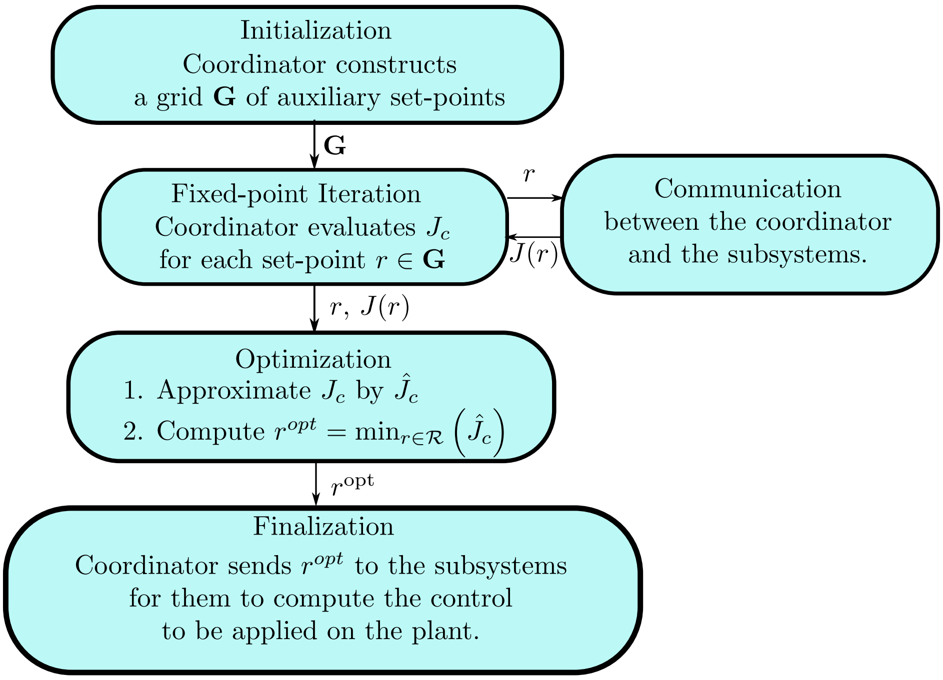


Figure 3.3: block diagram of the hierarchical control algorithm.

To conclude, Fig. 3.3 summarizes the whole hierarchical control algorithm.

3.4.3 Distributing the optimization over time

There are cases where the presence of nonlinearities and constraints, which are incoporated in the MPC problem, increase the computational burden. In other words, the computation of the n_{ev} necessary evaluations for central cost approximation might require a computation time that goes beyond the available sampling time T_s. This section proposes a method to reduce the computation time with a rather little impact on the quality of the resulting closed-loop performance. Since using constrained nonlinear MPC induces a significant increase in the computation time, it might be impossible to compute a solution r^{opt} (following the scheme of the previous section) for the next sampling period in the presence of limited computational resources. To overcome this potential issue, this section proposes a technique inspired by [44] which is based on the idea of distributing the optimization over time. In order to facilitate the following explanations, the notation k and $k+1$ are used to refer to instants $k\tau_u$ and $(k+1)\tau_u$ with τ_u being the control updating period, namely, the time during which the computation of a new optimal open-loop sequence is recomputed to implement the MPC feedback. Note that τ_u is not necessarily equal to the sampling time T_s. The process described in this section will be executed during the updating period $[k, k+1]$ as long as the computation time does not exceed τ_u.

Recall that the approximation of the cost function $J_c(r)$ needs the evaluation of J_c at $n_{ev} \geq (n_r+1)(n_r+2)/2$ values of the auxiliary set-points. By reducing the number of degrees of freedom (DOF) of vector r to be optimized from n_r to $n_z < n_r$, only $(n_z+1)(n_z+2)/2$ realizations would be needed, which accordingly leads to a decrease of the computation burden per updating period.

More precisely, a change in the decision variable is cyclically operated by defining a reduced dimensional parameterization of r of the form:

$$\tilde{r} = Mr + Dz \tag{3.43}$$

where $\tilde{r} \in \mathbb{R}^{n_r}$, $M \in \mathbb{R}^{n_r \times n_r}$ and $D \in \mathbb{R}^{n_r \times n_z}$. Moreover, the transformation matrices M and D are changed in a cyclic way in order to *explore* all the degrees of freedom of r after a finite number of successive iterations. This is explained in a more detailed way in the remainder of this section.

At the beginning of each updating period k, the optimization problem to be solved is given by:

$$z^\star(k) = \underset{z}{\operatorname{argmin}}\, \hat{J}_c(M^{(j_k)} r^\star(k-1) + D^{(j_k)} z) \tag{3.44}$$

where the transformation defined by the matrices $M^{(j_k)}$ and $D^{(j_k)}$ is defined in order to assign some components of the vector r to be equal to the corresponding components of the previous solution $r^\star(k-1)$ while leaving as degrees of freedom the n_z remaining components that define the reduced dimensional decision variable z. Note that the definition of the transformation matrices depends on the updating instant k through the upper index j_k, which is a cyclic variable defined by:

$$j_k = (j_{k-1} + 1) \mod n_r \tag{3.45}$$

In the numerical investigation, the following two configurations are tested in order to illustrate the proposed methodology:

Configuration 1: $n_z = 1, n_r = 3$

$$M^{(0)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \; D^{(0)} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.46}$$

$$M^{(1)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \; D^{(1)} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{3.47}$$

$$M^{(2)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \; D^{(2)} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.48}$$

Configuration 2: $n_z = 2, n_r = 3$

$$M^{(0)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \; D^{(0)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{3.49}$$

$$M^{(1)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \; D^{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.50}$$

$$M^{(2)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \; D^{(2)} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.51}$$

Note that the same methodology explained before regarding the definition of the grid of points is adopted with r and n_r respectively replaced by z and n_z. The only difference is that the number of degrees of freedom to be considered at the beginning of each updating period is reduced, and the significance of the degrees of freedom in terms of the components of r changes at each updating period.

When a sub-optimal solution $z^\star(k)$ to (3.44) is obtained (after the allowed number of iterations), the corresponding candidate sub-optimal auxiliary set-point $r_c^\star(k)$ is given by

$$\begin{aligned} r_c^\star(k) &= M^{(j_k)} r^\star(k-1) + D^{(j_k)} z^\star(k) \\ j_k &= (j_{k-1} + 1) \mod n_r \end{aligned} \tag{3.52}$$

This candidate value is then used to update the size of the trust region in a similar way as explained above. The method can be simply sketched by Algorithm 2 for a given updating cycle involving n_d iterations. More precisely, for-loop in Algorithm 2 allows to perform n_d iterations within the updating period. Indeed, if the computation time does not exceed the updating period $[kT_s, (k+1)T_s]$, the whole process mentioned in this section can be repeated in order to improve the sub-optimal candidate auxiliary set-point $r_c^\star(k)$.

Algorithm 2 Pseudo code for the distributed-in-time optimization

```
for $l \leftarrow 1, ..., n_d$ do
    Coordinator defines a grid of auxiliary set-points $\mathbf{G}(M^{(j_k)} r^\star(k-1) + D^{(j_k)}\Delta(\rho(k-1)), \mathcal{R})$;
    Coordinator evaluates the cost function for each element $r$ in the grid $\mathbf{G}(M^{(j_k)} r^\star(k-1) + D^{(j_k)}\Delta(\rho(k-1)), \mathcal{R})$;
    Coordinator computes the quadratic approximation $\hat{J}_c(z)$ of $J(z)$;
    Coordinator finds $z^\star(k)$ by solving (3.44)

    Coordinator computes the candidate auxiliary set-point $r_c^\star(k)$ according to (3.52);
    Coordinator updates $\rho$ and $r^\star(k)$ using (3.40) and (3.41);
end for
Coordinator sends $r^\star(k)$ to the subsystems;
```

3.5 Simulation results

3.5.1 Investigated system description and parameters setting

3.5.1.1 Description of the investigated system

The system that is chosen to validate the method is the cold box described in chapter 1. Fig. 3.4 shows a block diagram of the cold box system consisting of a Joule-Thomson cycle and a Brayton cycle. The Brayton cycle consists of two heat exchangers, which are NEF_2, NEF_{34} and a turbine T_1. The helium flow is cooled down using the cryogenic turbine T_1 to extract thermal energy from the fluid and by exchanging the heat power through a series of heat exchangers (NEF_x). A part of the helium gas is liquefied by the valve CV_{155} through the isenthalpic process and fall into the bath, while the gaseous part returns to the cycle via the cold pipeline.

The manipulated inputs and regulated outputs of this system are introduced as follows:

The Manipulated Inputs: There are three control inputs that are defined below:

1. $\text{CV}_{155} \in [0, 100]$ %: This valve is situated at the inlet of the helium bath.
2. $\text{NCR}_{22}^{(a)}$: This heating actuator is located inside the helium bath (S_1). The value of $\text{NCR}_{22}^{(a)}$ is in the range of $[0, 55]$ W. Note that the variable NCR_{22} in Fig. 3.4 is

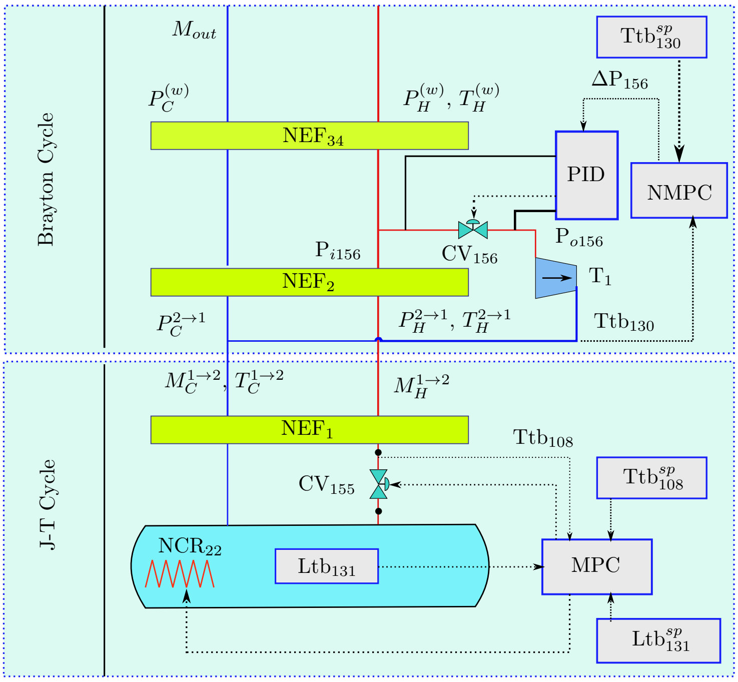


Figure 3.4: Block diagram of the cold box plant.

decomposed into two terms:

$$\mathrm{NCR}_{22} := \mathrm{NCR}_{22}^{(a)} + \mathrm{NCR}_{22}^{(w)} \tag{3.53}$$

where $\mathrm{NCR}_{22}^{(w)}$ represents the disturbance coming from the heat source.

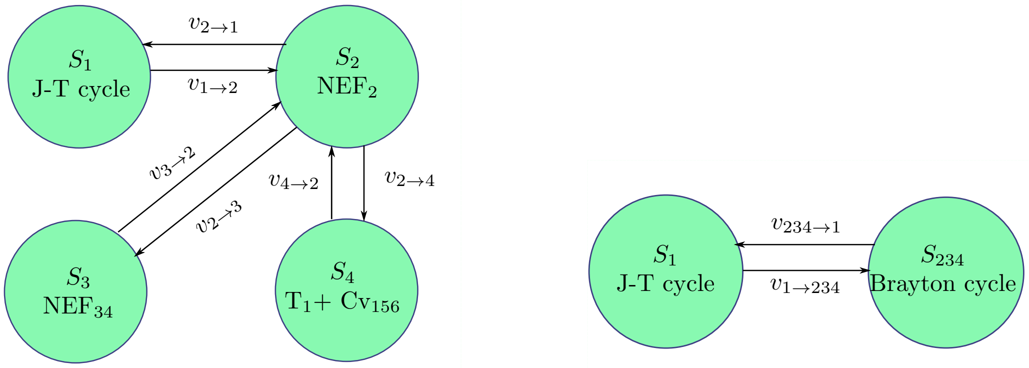
3. $\Delta\mathrm{P}_{156} \in [0, 12]$ bar: The pressure drop between the inlet pressure and outlet pressure of the valve CV$_{156}$. It should be noted that the valve CV$_{156}$ is used to control the pressure drop $\Delta\mathrm{P}_{156}$ between its inlet and outlet pressure. To do so, the local NMPC of the turbine T$_1$ computes and sends an appropriate value of the pressure drop $\Delta\mathrm{P}_{156}$ to the PID controller, which acts on the opening position of the valve CV$_{156}$ (Fig. 3.4). This PID controller is used to hide the nonlinearity of the valve CV$_{156}$.

The Regulated Outputs: There are three regulated outputs and one constrained output (see Figure 3.4. for the notation):

1. Ltb$_{131}$: The helium liquid level (%). The set-point is chosen by the operator. In the usual operation, it is set at $\mathrm{Ltb}^{sp}_{131} = 60.5$ %.
2. Ttb$_{108}$: The temperature at the inlet of the J-T valve must be tightly controlled in order to ensure the efficiency of the liquefaction of the helium. The setpoint for this output is $\mathrm{Ttb}^{sp}_{108} = 5.24\ K$.
3. Ttb$_{130}$: The temperature at the turbine T$_1$'s outlet. The setpoint for this output is $\mathrm{Ttb}^{sp}_{108} = 12.23\ K$.

4. M_{out}: The output flow rate that enters the warm compression station. This flow rate M_{out} is not regulated but is limited below 70 g/s, which is the allowable flow rate that can be handled by the warm compression station.

In Chapter 2, it has been shown that the cold box can be decomposed into either two subsystems or four subsystems. The subsystems of these two decomposition strategies are indexed as shown in Fig. 3.5 and described below:



(a) (b)

Figure 3.5: Two possible decomposition of the Cold box: 4-subsystem topology (a) and 2-subsystem topology (b).

Four-subsytem topology (4ss strategy): This decomposition consists of the Joule-Thomson cycle (S_1), the heat exchanger NEF$_2$ (S_2), the heat exchanger NEF$_{34}$ (S_3) and the turbine T_1 (S_4). In this network, the turbine employs a nonlinear static model while the heat exchangers and the Joule-Thomson cycle employ linearized dynamic models. With a slight abuse of notation, x_s, u_s, $v_{s\to s'}$, and y_s are used to represent respectively the deviations from the operating points x_s^{op}, u_s^{op}, $v_{s\to s'}^{op}$ and y_s^{op}. Note that only the turbine T_1 and Joule-Thomson cycle are controlled by NMPC and MPC, respectively, while the other subsystems are impacted by their decisions. Finally, Their mathematical models are listed below:

Subsystem S_1: The Joule-Thomson cycle:

$$x_1^+ = A_1 x_1 + B_1 u_1 + \sum_{s'\in\mathcal{N}_1} G_{s'\to1} v_{s'\to1} + F_1 w_1, \tag{3.54}$$

$$y_1 = C_1 x_1, \tag{3.55}$$

$$v_{1\to s'} = C^v_{1\to s'} x_1, \quad \forall s' \in \mathcal{N}_1 \tag{3.56}$$

where $x_1 \in \mathbb{R}^{30}$ is the state vector of subsystem S_1 and $w_1 = [\mathrm{NCR}_{22}^{(w)}]$ indicate the disturbance vector. The output vector and manipulated input vector are respectively represented by $y_1 = [\mathrm{Ltb}_{131}, \mathrm{Ttb}_{108}]^T$ and $u_1 = [\mathrm{CV}_{155}, \mathrm{NCR}_{22}^{(a)}]^T$.

Subsystem S_2 and S_3 : The heat exchangers NEF_2 and NEF_{34}, respectively:

$$x_s^+ = A_s x_s + \sum_{s' \in \mathcal{N}_s} G_{s' \to s} v_{s' \to s} \quad \forall s \in \{2,3\},\ s' \in \mathcal{N}_s \tag{3.57}$$

$$y_s = C_s x_s, \quad \text{for } s = 3,\ s' \in \mathcal{N}_s \tag{3.58}$$

$$v_{s \to s'} = C^v_{s \to s'} x_s \quad \forall s \in \{2,3\},\ s' \in \mathcal{N}_s \tag{3.59}$$

where $x_2 \in \mathbb{R}^{34}$ and $x_3 \in \mathbb{R}^{32}$ are the state vectors of subsystem S_2 and S_3, respectively. Note that these two subsystems do not have any control input while their dynamic is affected by other subsystems' coupling signals $v_{s' \to s}$. Concerning the outputs, the subsystem S_3 has one which is $y_3 = M_{out}$. Since the subsystem S_2 does not have any output, its related equation y_2 does not exist.

Subsystem S_4 : Turbine T_1:

$$y_4 = h_4(u_4, v_4^{in}) \tag{3.60}$$

$$v_{4 \to s'} = g_4(u_4, v_4^{in}), \quad s' \in \mathcal{N}_4 \tag{3.61}$$

where $v_4^{in} := \text{col}_{s' \in \mathcal{N}_4}(v_{s' \to 4})$. The output vector and manipulated input vector are respectively $y_4 = \text{Ttb}_{130}$ and $u_4 = \Delta\text{P}_{156}$. Note that the subsystem S_4 is only a static function of control input u_4 and the incoming coupling signal v_4^{in}.

Two-subsystems-decomposition (2ss strategy): This decomposition consists of two subsystems that are the Joule-Thomson cycle (S_1) and the Brayton cycle (S_2) as already depicted in Fig. 3.4. Note that the turbine T_1 and the two heat exchangers (NEF_2 and NEF_{34}) are now combined to become a larger subsystem S_{234} that represents the Brayton cycle. These two subsystems, in this decomposition, are regulated, which follows the assumption made in previous work [1]. The model equation of Joule-Thomson cycle is kept unchanged, while the state-space representation of the Brayton cycle can be easily obtained by combining the equations of S_2, S_3 and S_4, which yields the following ***nonlinear*** model:

$$x_{234}^+ = f_{234}(x_{234}, u_{234}, w_{234}, v_{234}^{in}) \tag{3.62}$$

$$y_{234} = h_{234}(x_{234}, u_{234}) \tag{3.63}$$

$$v_{234 \to 1} = g_{234}(x_{234}, u_{234}) \tag{3.64}$$

where

$$x_{234} = [x_2^T, x_3^T]^T; \qquad u_{234} = u_4; \qquad w_{234} = w_3; \tag{3.65}$$

$$v_{234}^{in} = v_{1 \to 2}; \qquad y_{234} = [y_4^T, y_3^T]^T; \qquad v_{234 \to 1} = v_{2 \to 1} \tag{3.66}$$

are states, control inputs, disturbance input, incoming coupling signal, regulated outputs and outgoing coupling signals with appropriate dimensions of the new subsystem, respectively.

⊳ **Important**

It is essential to note that in this new subsystem S234, the nonlinearity of the turbine makes the whole model become a large-scale nonlinear system despite the fact that the heat exchanger parts are linear.

Table 3.1: The inputs, outputs and the coupling variables of the 4-subsystems topology and the 2-subsystems topology. The notations T_L, M_L and P_L / T_H, M_H and P_H represent the temperature, the flow rate and the pressure of the low pressure / high pressure pipeline, respectively.

4-subsystem topology				
Subsystem	u_s	w_s	y_s	$v_{s \to s'}$
S_1	$\text{NCR}_{22}^{(a)}$ CV_{155}	$\text{NCR}_{22}^{(w)}$	Ltb_{131} Ttb_{108}	$v_{1\to 2} = [M_H^{1\to 2}, M_L^{1\to 2}, T_L^{1\to 2}]^T$
S_2	–	–	–	$v_{2\to 1} = [T_H^{2\to 1}, P_H^{2\to 1}, P_L^{2\to 1}]^T$ $v_{2\to 3} = [M_H^{2\to 3}, M_L^{2\to 3}, T_L^{2\to 3}]^T$ $v_{2\to 4} = [P_C^{2\to 4}]$
S_3	–	–	M_{out}	$v_{3\to 2} = [T_H^{3\to 2}, P_H^{3\to 2}, P_L^{3\to 2}]^T$ $v_{3\to 4} = [T_H^{3\to 4}, P_H^{3\to 4}]^T$
S_4	ΔP_{156}	–	Ttb_{130}	$v_{4'\to 2'} = [M_L^{4\to 2}, T_L^{4\to 2}]^T$ $v_{4\to 3} = [M_H^{4\to 3}]$
2-subsystem topology				
Subsystem	u_s	w_s	y_s	$v_{s \to s'}$
S_1	$\text{NCR}_{22}^{(a)}$ CV_{155}	$\text{NCR}_{22}^{(w)}$	Ltb_{131} Ttb_{108}	$v_{1\to 2} = [M_H^{1\to 2}, M_L^{1\to 2}, T_L^{1\to 2}]^T$
S_{234}	ΔP_{156}	–	$\text{Ttb}_{130}^{(a)}$ M_{out}	$v_{2\to 1} = [T_H^{2\to 1}, P_H^{2\to 1}, P_L^{2\to 1}]^T$

To summarize, Table 3.1 shows the inputs u_s, outputs y_s, disturbance inputs w_s, and the coupling signal $v_{s\to s'}$ of these two decomposition strategies.

3.5.1.2 Parameters setting

Central cost parameters: The role of the coordinator is to control the system through two operation modes which are recalled hereafter:

1. In the first mode, the objective is to regulate the system around the nominal point $x = 0$

in spite of the unmeasured disturbances (disturbance-rejection mode).

2. In the second mode, the coordinator can temporarily drive the system to a different steady-state corresponding to a new set-point $y \neq 0$. For instance, the operator might decide to change the level Ltb_{131} of liquid helium in the bath or the temperature Ttb_{108}. This corresponds to a change in the corresponding set-points.

These two modes can be taken into account by using different weighting matrices in the local cost that contributes to the central cost function. These local costs will be introduced hereafter.

For S_1 and S_4 that need to track the desired set-points r_s^d:

$$J_s(r|r_s^d) = \sum_{i=0}^{N-1} \|y_s(k+i) - r_s^d\|^2_{Q_c^{(s)}} + \|u_s(k+i)\|^2_{R_c^{(s)}} \tag{3.67}$$

where $Q_c^{(s)}$ and $R_c^{(s)}$, for $s \in \{1,4\}$ are chosen to be positive semidefinite matrices, namely:

✓ **Mode 1**: For disturbance rejecting scenario:

$$Q_c^{(1)} = \begin{bmatrix} 10^4 & 0 \\ 0 & 10^4 \end{bmatrix}, \qquad R_c^{(1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.68}$$

$$Q_c^{(4)} = 10^4, \qquad R_c^{(4)} = 0 \tag{3.69}$$

✓ **Mode 2**: For set-point tracking scenario:

$$Q_c^{(1)} = \begin{bmatrix} 10^6 & 0 \\ 0 & 0.1 \end{bmatrix}, \qquad R_c^{(1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.70}$$

$$Q_c^{(4)} = 10^4, \qquad R_c^{(4)} = 0 \tag{3.71}$$

For S_3 that has output M_{out} to be constrained, the constraint violation cost is defined, namely:

$$J_3(r|\overline{y}_3) = \sum_{i=0}^{N-1} \|\max(y_3(k+i) - \overline{y}_3, 0)\|^2_{Q_{cstr}^{(3)}} \tag{3.72}$$

where $\overline{y}_3 = 0.07\ kg/s$ and $Q_{cstr}^{(3)} = 10^9$ are fixed in the two mentioned modes.

For S_2 that does not have any contribution to the central cost, its cost is simply defined by $J_2(r) = 0$.

The local costs for the 2-subsystems topology can simply be deduced from the previous choices so that the resulting central cost is identical.

! Attention

It is essential to note that these local costs are different from the control costs implemented by the local NMPC/MPC of the subsystems, which are presented hereafter.

Local MPC and NMPC parameters: Recall that the agents implement MPC (for S_1) and NMPC (for S_4) in order to control their corresponding local subsystems. The penalty matrices for these local MPCs/NMPCs are fixed as below:

$$Q_1 = C_1^T \cdot \begin{bmatrix} 10 & 0 \\ 0 & 100 \end{bmatrix} \cdot C_1 \qquad R_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$Q_4 = 10^3 \qquad R_4 = 1$$

Similarly, the weighting matrices for the NMPCs/MPCs in 2-subsystems-topology can be deduced by basing on the above choices. More precisely, the weighting matrices for S_1 Q_1 and R_1 are kept unchanged, while Q_{234} and R_{234} are defined as below:

$$Q_{234} = C_{234}^T \cdot Q_4 \cdot C_{234} = C_{234}^T \cdot 10^3 \cdot C_{234} \qquad R_{234} = 1; \tag{3.73}$$

where the matrix C_{234} is the partial derivative of $h_{234}(x_{234}, u_{234})$ in (3.63) with respect to x_{234} at an operating point (x_{234}^{op}, u_{234}).

> Important

In contrast to the central penalty matrices $Q_c^{(s)}$, $Q_{cstr}^{(s)}$, $R_c^{(s)}$, which can be modified freely according the different operation modes, the local penalty matrices Q_s and R_s are hard-fixed and are not allowed to be changed during their operation.

The prediction horizon N is chosen to have a length of NT_s (where $N = 100$ and $T_s = 5$ s). This corresponds roughly to 8 minutes. This setting is currently used at CEA/IRIG/DSBT and is also the one that has been used in many previous studies involving MPC control design.

Fixed-point iteration parameters: The termination criteria on convergence error ϵ_{max} and the maximum number of iteration described in section 3.3.1 are set at $\epsilon_{max} = 10^{-5}$ and $\sigma_{max} = 30$, respectively. For the updating rules of the trust region size, the parameters β^-, β^+ are set to 0.7 and 1.25, respectively.

Performance index: Indeed, a cost index is necessary in order to evaluate and compare the performances in terms of closed-loop costs associated to the different framework settings. The commonly used closed-loop central cost will be adopted, namely:

$$J_{sim} = \frac{1}{N_{sim}} \sum_{s\in\mathcal{N}} \sum_{i=1}^{N_{sim}} \Big[\|y_s^{sim}(i) - r_s^d(i)\|_{Q_c^{(s)}} + \|u_s^{sim}(i)\|_{R_c^{(s)}} \tag{3.74}$$

$$+\| \max(y_s^{sim}(i) - \overline{y}_s, 0)\|_{Q_{cstr}^{(s)}} \Big] \tag{3.75}$$

where N_{sim} is the length of the simulation (in terms of sampling periods $T_s = 5$ s). The regulated outputs $y_s^{sim}(i)$ and manipulated inputs $u_s^{sim}(i)$ are system behavior obtained during a simulation under the presented configurations and the two specific system's decompositions. The weighting matrices $Q_c^{(s)}$, $Q_{cstr}^{(s)}$ $R_c^{(s)}$ are the penalty matrices previously defined in the local costs (3.67)-(3.72).

In addition, in order to supervise the convergence of the processed fixed-point iterations, the maximum terminal error is monitored at every time step, namely:

$$\overline{\epsilon}(k) := \max_{i=1}^{n_{ev}}(\epsilon_{end}^{(i)}) \tag{3.76}$$

where $\epsilon_{end}^{(i)}$ ($\forall i \in \{1,\dots,n_{ev}\}$) are the terminal convergence errors resulted from every central cost evaluation.

State observation: In order to estimate the states x_s to be used for updating the MPC/NMPC's laws, the centralized observer described in the last chapter is implemented. The observer not only estimates the states x_s but also estimates the exogenous inputs v_s (extended observer), which is used when the hierarchical coordination is not activated. More precisely, the exogenous signals $v_s(k)$ are supposed to remain constant over the prediction horizon in the decentralized settings (without the coordinator). However, it should be noted that if each local subsystem has output measurements, which makes it observable, the decentralized observation strategy can be made, which means that each of them can have its own observer to estimate their states.

3.5.2 Numeric simulation results:

In this section, the decomposition strategies that are mentioned previously are employed to conduct a series of simulations in order to verify several aspects.
The 2-subsystem-decomposition strategy will be employed to validate:

- The benefit from using nonlinear models,
- The benefit from hierarchical design,
- Impact of the distributed-in-time setting's parameters,

- Checking modularity: controlling the system by only tuning the central cost's definition.
- Impact of 4-subsystem-decomposition strategy on computation time.

The 4-subsystem-decomposition strategy will be employed to validate the impact of 4-subsystem-decomposition strategy on the computation time and the control performance.

3.5.2.1 The benefit from using nonlinear models

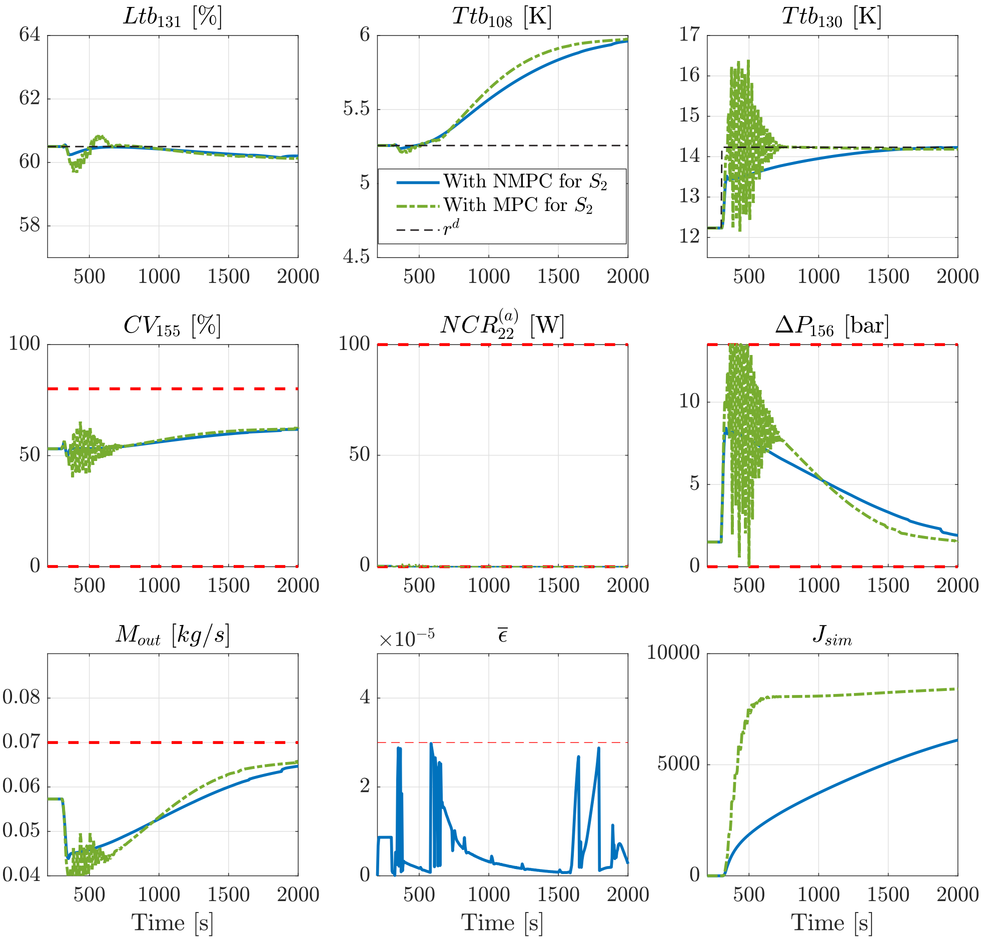


Figure 3.6: Closed-loop behaviors when using the hierarchical approach with nonlinear MPC (solid blue line) and with linear MPC (dash-dot green line). The first row presents the outputs, and the second one presents the inputs of the system.

To start, the set-point tracking on Ttb_{130} scenario is conducted. As illustrated in Fig. 3.6, the set-point of Ttb_{130} (subfigure(1,3)) is changed to be far from the operation point by which the Brayton cycle model (S_{234}) is linearized. Consequently, the system behavior given by the coordination with linear MPC is highly oscillatory, while the one with nonlinear MPC is stabilized. In addition, subfigure(3,2) also shows that the convergence error is assured below the defined limit $\epsilon_{\max}$.

3.5.2.2 The benefit from hierarchical design

In this section, the disturbance rejection mode is simulated. The closed-loop cost of the proposed hierarchical framework is compared to the one obtained under the extended observer-based decentralized approach. More precisely, two scenarios , in which the limitation on the available computation time is respectively enforced or not, are simulated with the disturbance profile (Fig. 3.7) .

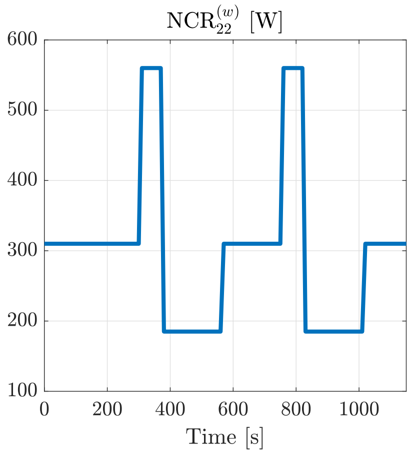


Figure 3.7: Disturbance profile of $\mathrm{NCR}_{22}^{(w)}$

- In the scenario where the constraint on computation time is not taken into account, the corresponding $u_s^{opt}(k)$ is implemented even if its computation time exceeds the available time within the sampling period.
- Meanwhile, in the scenario where the cpu-constraint ($\tau_u \leq T_s$) on the available computation time is enforced, the subsystems apply the previous control $u_s^{\text{opt}}(k-1)$ each time the computation time exceeds the available computation time T_s. Indeed, in this case, the master cannot dispose of the needed information in order to update the approximation of the cost function, which is needed to update the value of the auxiliary set-point and the associated coupling signals that are needed to compute the updated control to be applied.

Fig. 3.8 shows the system behavior of the two scenarios, comparing with the decentralized control approach. It shows that without taking into account the computation time constraint, the coordinator drives the system so that the flow rate M_{out} (subfigure(3,1)) returns to be lower than the acceptable value $\overline{M}_{out}$, in this case, the valve CV$_{156}$ is closed in order to decrease the fluid passing through it. However, using nonlinear MPC in the proposed framework increases the computation burden since the optimal control problems are solved repeatedly in the fixed-point iteration. Probably, the coordination performance worsens if the constraint on the computation time is enforced, which is shown by the non-stabilized regulated outputs due to non-appropriately updated control. Finally, the subfigure(3,3) compares the performance index J_{sim} of the scenarios, which shows that the control under coordination gives better performance compared to the control without coordination. Nevertheless, it also shows that the performance of the coordination can be worse if the computation time constraint is considered.

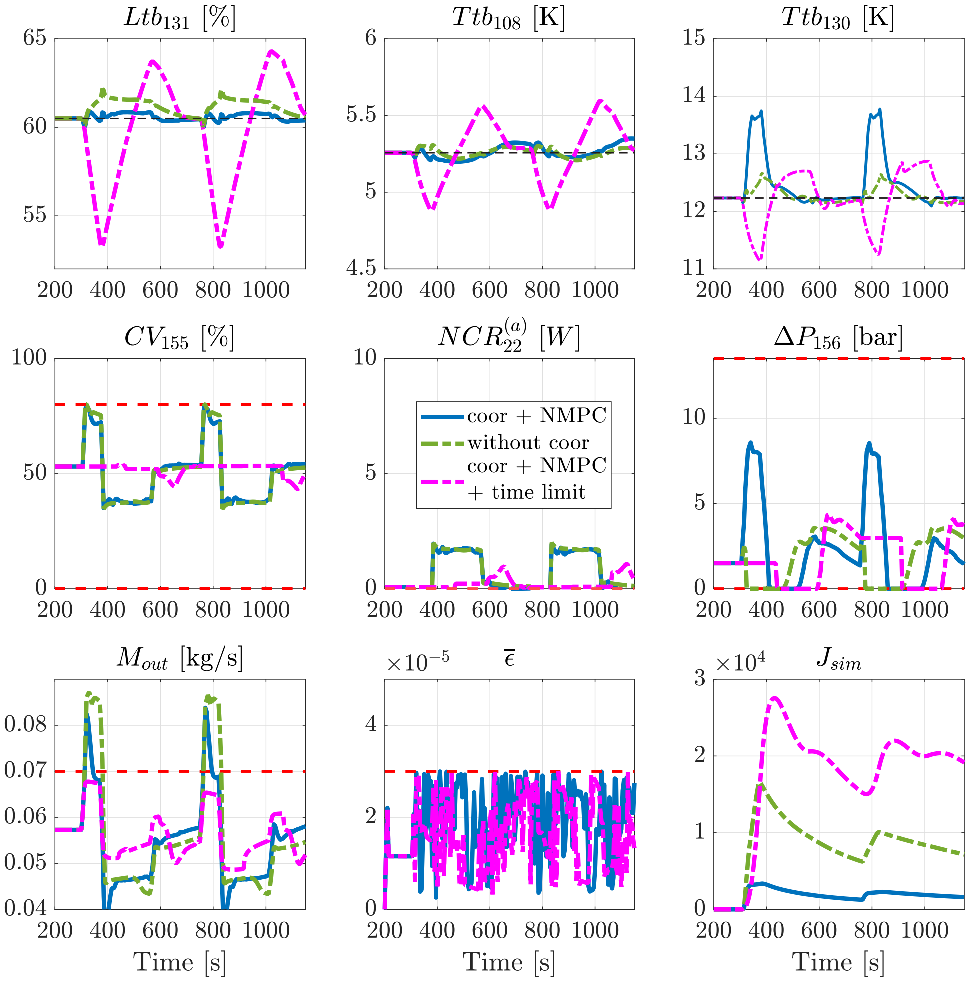


Figure 3.8: Closed-loop behavior with ideal coordination (solid blue line), without coordination (dash-dot green line), and coordination in taking into account the computation limit (solid purple line). The first row presents the outputs, and the second one presents the inputs of the system. The hierarchical control method gives a better cost J_{sim} than the decentralized method (closed-loop cost decreased by 56%). Moreover, note in particular how the control of the liquid helium level Ltb$_{131}$ visibly deteriorates when the computation time limit is considered.

Fig. 3.9 shows the associated histogram of the computation time of the subsystems S_1 and S_{234}. It can be realized that the computation time of S_{234} exceeds the control updating period $[k, k+1]\tau_u$ (with $\tau_u = 5s$).

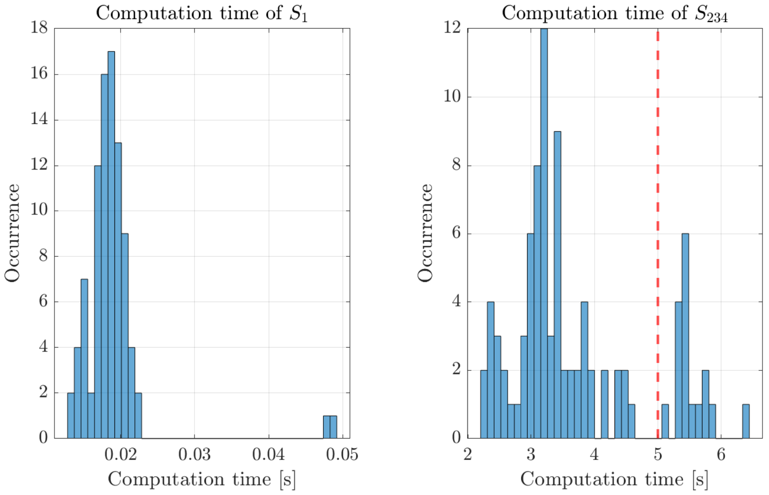


Figure 3.9: Histogram of the computation time of the JT cycle S_1 and the Brayton cycle S_{234}.

In the next section, the possibility to partially recover the optimal performance through the distributed-in-time optimization scheme proposed in section 3.4.3 is investigated.

3.5.2.3 Impact of the distributed-in-time setting's parameters

Different configurations of the distributed-in-time optimization parameters (n_z, n_d and τ_u) are simulated, and the corresponding closed-loop costs $J_{\text{sim}}^{\text{norm}}$ (normalized cost index) are reported in order to give a flavour of the impact of each choice on the results. More precisely, the testing scenario of a periodic heating disturbance (Fig. 3.7) is simulated again with the distributed-in-time optimization framework being implemented in the hierarchical framework.

Fig. 3.10 shows the comparison of the behaviors of the process in three different scenario: 1) using the hierarchical control combined with optimization distribution, 2) with coordination (not taking into account the time limit constraint) and 3) without coordination. It shows that the optimization distribution technique can give a similar performance compared to the one given by the ideal coordination (without taking into account time limit constraint) (see subfigure(3,3)).

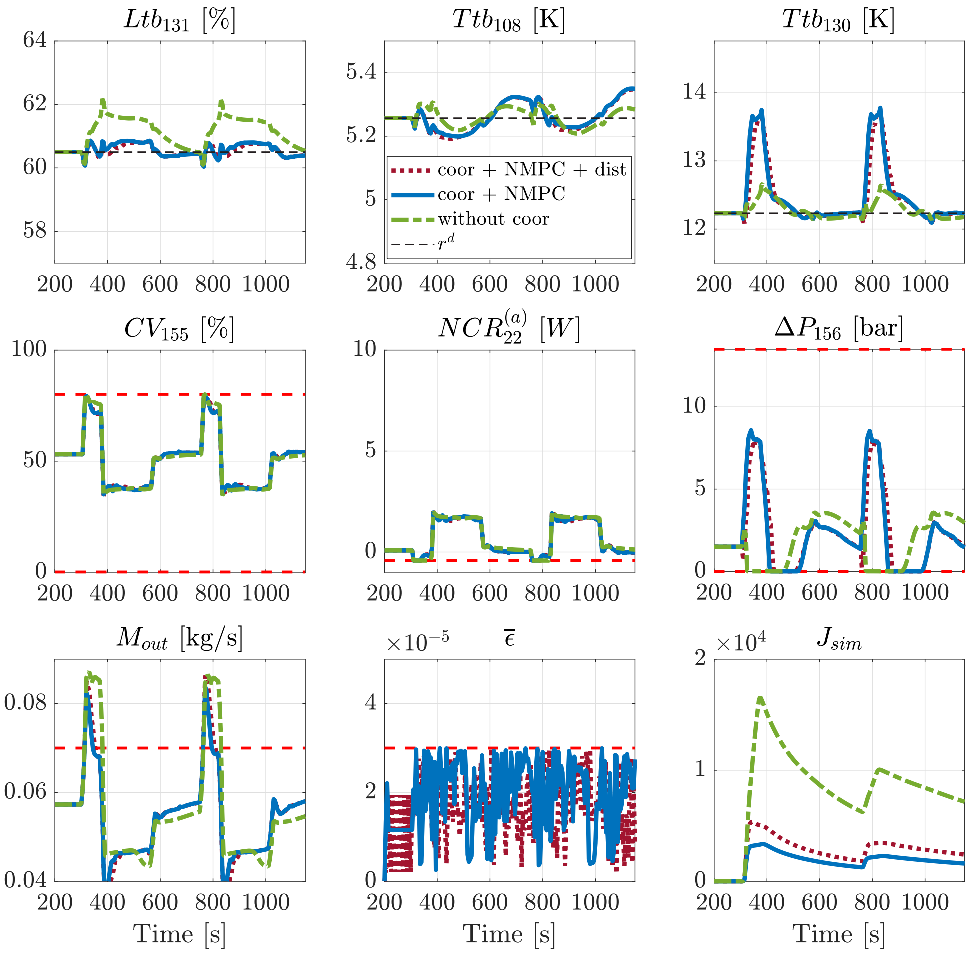


Figure 3.10: Comparison of the closed-loop behavior under the proposed hierarchical framework with two different settings: 1) with distributed-in-time optimization being implemented, 2) with coordination but not taking into account the time limitation constraint and 3) without coordination. The first row presents the outputs and the second one presents the inputs of the system. The choice of parameters $n_z = 2$, $n_d = 1$ and $\tau_u = 5s$ are used in the distributed-in-time optimization framework.

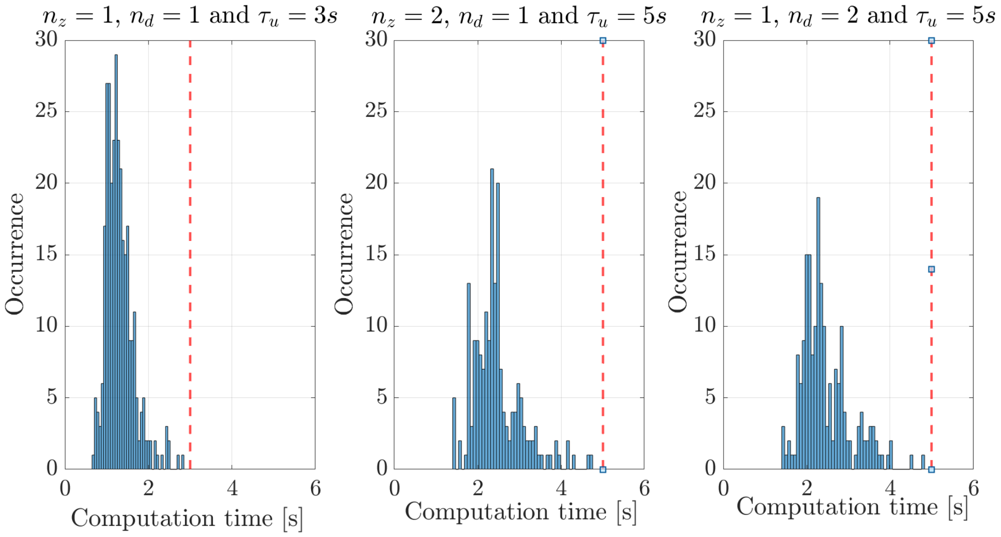


Figure 3.11: Histogram of computation time of different choices of n_z and n_d. Note that the computation time of each configuration is always lower than the updating time τ_u.

Fig. 3.11 shows the computation time for different configuration of n_z and n_d. By using the optimization distribution technique with the appropriately chosen parameters n_z and n_d, the feasibility of the whole framework is ensured, which means the computation time is lower than the updating time constant τ_u.

Finally, table 3.2 shows the associated closed-loop cost for the different configurations of the distributed-in-time optimization scheme. It comes out that the setting corresponding to $n_z = 2$, $n_d = 1$ enables to get closer to the ideal cost index drop (44%) corresponding to the non constrained computation time simulation while being fully real-time compatible.

Table 3.2: The normalized cost J_{sim}^{norm} for different configurations n_z, n_d and τ_u of the distributed-in-time optimization.

	n_z	n_d	τ_u	t_{max}	$J_{\text{sim}}^{\text{norm}}$
Decentralized method	_	_	_	_	100%
Without distribution (with computation time limit)	_	_	_	_	270.12%
Hierarchical method With distribution	1	1	3s	2.85s	33.97%
	1	2	5s	4.84s	33.99%
	2	1	5s	4.76s	33.7%
Without distribution (without computation time limit)	_	_	_	_	22.45%

3.5.2.4 Checking modularity: controlling the system by only tuning the central cost's definition

One of the claims of this contribution concerns the possibility of keeping the local controllers unchanged (in terms of penalty) while changing the penalties of the central cost (by the coordinator) in order to achieve different behaviors of the closed-loop system. Here, it is assumed that the operator needs to change the set-point of the helium liquid level Ltb$_{131}$, for example, in order to embed a test facility below the liquid level. Thus, the reference tracking scenario is simulated. More precisely, the closed-loop behavior is compared under the two different modes defined in section 3.5.1.2. This is done in order to illustrate the fact that the results can be affected in the desired direction by only modifying the central cost's definition while keeping the local controllers unchanged.

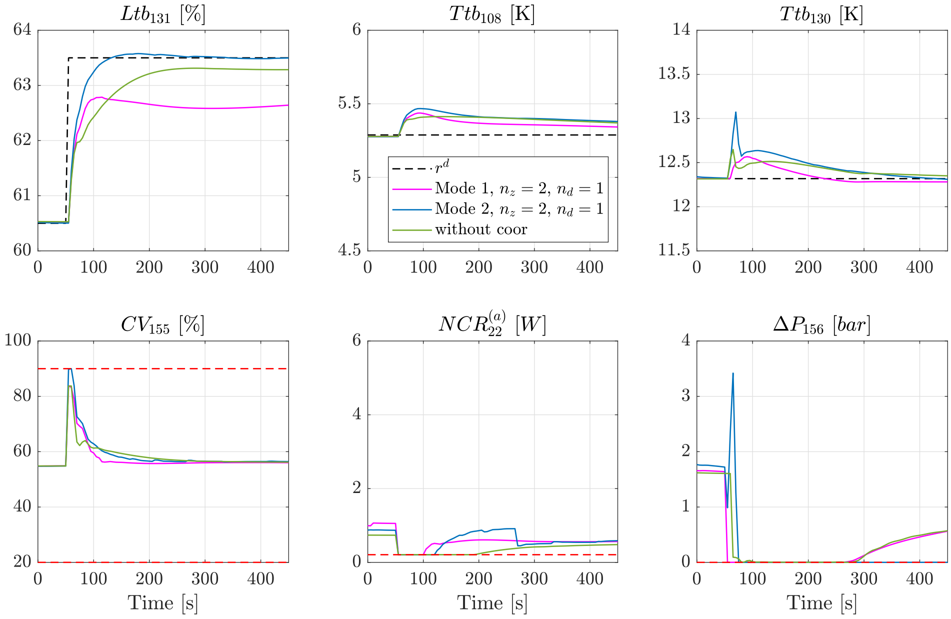


Figure 3.12: Set-point tracking scenario: closed-loop responses under coordination, using distributed-in-time optimization in two different mode for the centralized cost on one hand and without coordination on the other hand. The first row presents the outputs, and the second one presents the inputs of the system. The set-point on Ltb_{131} is increased. Two configurations of Q_c and R_c of Mode 1 and Mode 2 are tested. Mode 2 (corresponding to higher penalty on Ltb$_{131}$ deviations) allows better reference tracking while mode 1 which is dedicated to disturbance rejection and not especially to track set-point on the level. With the set of parameters $n_z = 2$, $n_d = 1$ and $\tau_u = 5s$. Note that both hierarchical design with distributed optimization are real-time compatible.

For the tracking set-point mode, Fig. 3.12 shows the comparison of the behavior of the process between using hierarchical control combined with optimization distribution and using decentralized control.

3.5.2.5 Impact of 4-subsystem-decomposition strategy on computation time

In this section, we will verify whether the application of the proposed framework is valid for the case where some of the subsystems are not controlled. More precisely, the coordinator will be applied on the same cryoplant (cold box) that is decomposed into four subsystems: J-T cycle (S_1), two heat exchangers (S_2 and S_3), and turbine T_1 (S_4) as described in 3.5.1.1. Recall that in this decomposition, the Joule-Thomson cycle and the turbine T_1 are respectively controlled by MPC and NMPC, while the two heat exchangers are not regulated but impacted by their neighbor dynamics.

Fig. 3.14 compares the two decomposition strategies (2-subsystem-decomposition and 4-subsystem-decomposition) in the same disturbance scenario; It can be seen that the two strategies give similar performance indexes.

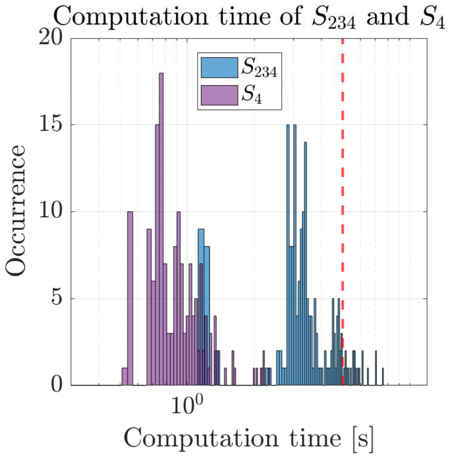


Figure 3.13: Computation time of S_{234} and S_4

Fig. 3.13 shows the computation time of the subsystems that implement the relative nonlinear MPC. It shows that the 4-subsystems decomposition strategy is implementable in real-time implementation, while the other is not implementable (without considering distributing optimization in time), which has been shown previously. This demonstration showed the ability of the proposed framework to solve a bottleneck problem where the nonlinearity attached to larger entity results in a computational burden.

3.6 Conclusion

This chapter presents a recently proposed hierarchical control framework based on fixed-point iteration with an application to the control of a cryogenic refrigerator. It is successfully demonstrated that incorporating nonlinearity and constraints into local controls can be an

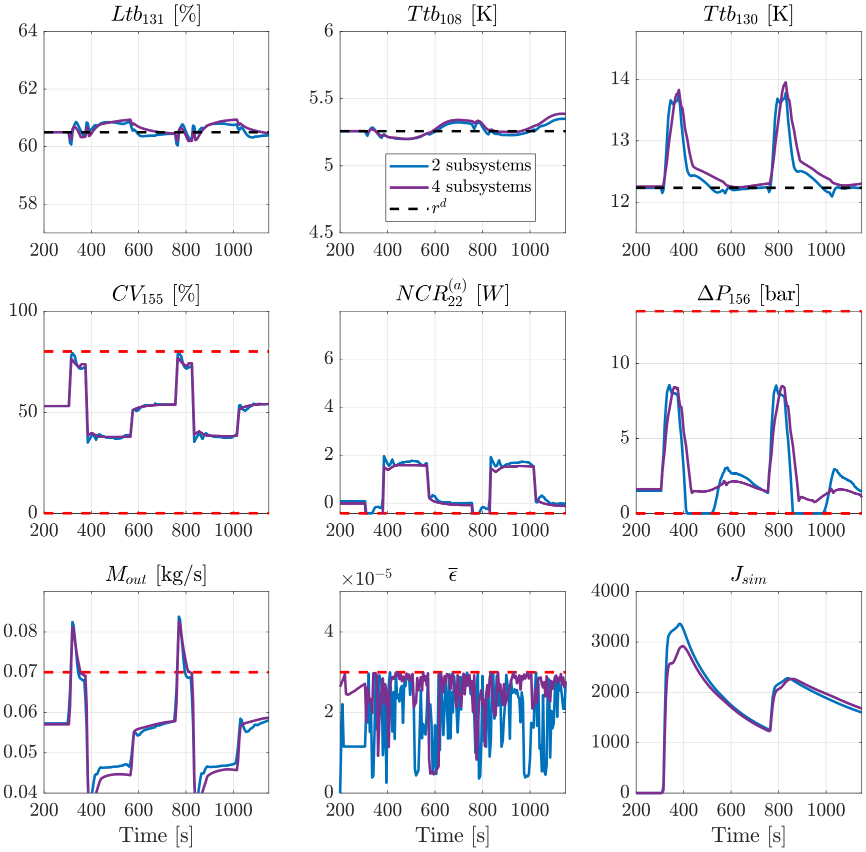


Figure 3.14: Comparison of the closed-loop behavior under the proposed hierarchical framework with two different strategies: 1) with 2-subsystem-decomposition strategy and 2)with 4-subsystem-decomposition strategy. The same disturbance profile on $\mathrm{NCR}_{22}^{(w)}$ that is used in the previous simulations is employed.

effective way to improve control quality. Furthermore, it is also shown that (in this example) locally proven convergence of the fixed point iteration is assured with such incorporation. Furthermore, a dedicated optimization distribution technique is proposed and validated, which allows to recover a large loss of optimality induced by the infeasibility of the control algorithm in a limited computational time. It is also shown the choice of the decomposition might help isolating the nonlinearity into small subsystem and reducing the bottleneck in terms of CPU.

Chapter 4

On the use of fast-NMPC and deep learning approach in fixed-point-based hierarchical control

---

Abstract The last chapter described a framework based on fixed-point iteration. This framework is dedicated to the control of a network of interconnected subsystems such as those describing cryogenic processes or power plants, by coordinating local controllers to optimize a global objective. Previous results have shown that dealing with constraints and nonlinearities can challenge the real-time feasibility of the approach. This chapter investigates and combines two interesting directions, namely the use of truncated fast gradient and deep neural network based controller modeling, to reduce the computation time of the most critical subsystem. It is also shown that by doing so, the control updating period can be significantly reduced and the closed loop performance can be greatly improved. This chapter can therefore be seen as a concrete implementation and validation of some key ideas in the design of real-time distributed NMPCs. All concepts are validated using the realistic and challenging example of a real cryogenic refrigerator.

4.1 Introduction

The last chapter showed that incorporating nonlinearities into local control problems can improve the performance of the overall framework. However, it can lead to losses of control performance if the computation becomes too complex and infeasible within the allowed computation time due to limited computational resources. This fact has been demonstrated by numerical simulations with the use of NMPC for the Brayton cycle in Chap. 3. In fact, the method consisting in distributing the optimization over the real-life time is only valid if the computation time resulted from the distribution is compatible with some predefined updating periods $[k, k+1]\tau_u$. Furthermore, finding a finer decomposition would be not a feasible choice if the aimed nonlinearity is not condensed and not separable from the larger subsystem.

One can realize that the computational burden is due to the resolution of local nonlinear MPC problems being repeated over the fixed-point iterations and for several set-points.

This computational bottleneck is induced by using powerful but computationally expensive solvers such as Casadi [45] or Acado [46]. Based on this observation, this chapter proposes two directions that could be used if the computational time problem is encountered when implementing the proposed framework.

In order to replace such powerful but non-integrable optimization solvers, the simplest way is to use a sub-optimal solver. Indeed, in [47], a well-known gradient-based iterative solver is proposed to solve linear optimization control problem, by providing a technique to define lower iteration bound. This has prompted much works regarding the implementation aspect of MPC in embedded applications [48, 49, 50].

Another way to reduce the computation time is to approximate the control laws by piecewise affine functions (PWA) defined on a polyhedral partition of the feasible states [51, 52, 53]. This method is also called explicit MPC. However, this property is only true if there is no nonlinearity present in the objective function or constraints. Besides, the complexity of the state space regions over which the control law is defined grows exponentially as the number of states increases, which makes this approach impractical for large-scale systems. Moreover, in the conventional explicit MPC method, only the first action of the control sequence is approximated, whereas, in our proposed framework, the entire control sequence $\boldsymbol{u}_s$ is required for the fixed-point iteration.

Instead of approximating the nonlinear MPC by PWA functions, deep learning has become a popular choice due to its universal approximation property. Moreover, many works have demonstrated the effectiveness of these methods in many embedded applications [54, 55, 56, 57]. Hence, this section attempts to address the computation time issue by following the track described below:

- First, a fast gradient-based algorithm is proposed to be used instead of using the time-consuming one in order to reduce the computation time. The performance of this solver is compared to available generic toolkits (such as Casadi/IPOPT). This solver will then be shown to be integrable into the fixed-point-based hierarchical control framework, making the whole algorithm feasible in real-time implementation.
- For embedded applications, deep learning approach has been demonstrated in several studies. More precisely, a feed-forward deep neural network is used to approximate the control laws computed by the proposed solver to further reduce the computation time of the most CPU-critical local controller.
- Finally, we will show that the reduction in computation time allows the control inputs to be updated more frequently, thus improving the closed-loop performance.

The next section will describe the Truncated gradient-based algorithm that is used to solve nonlinear optimal control problems.

4.2 Truncated gradient method for solving NMPC problem

In the previously described framework, the resolution of the local NMPC problems is proceeded in parallel, the real-time feasibility of the framework depends on the critical subsystem requiring the highest computation efforts. Although many toolkits for solving optimization-based control problems, such as ACADO [46] or CasADi [45] are widely used because of their ease of implementation, the relative complexity of the solvers associated to these toolkits can exceed the available computational resources. Furthermore, it has been shown in [58] that when a limited (computation time)/(hardware performance) is present, a truncated fast gradient might be beneficial to closed-loop performances. That is why this algorithm is briefly recalled here as it is the heart of the forthcoming development.

Recall that each subsystem S_s, $s \in \mathcal{N}^{ctr}$ solves an optimization problem upon receiving a pair of $(r_s, \boldsymbol{v}_s^{in})$ from the coordinator, combining with the estimated state $\hat{x}_s$ and the disturbance profile $\boldsymbol{w}_s$:

$$\mathcal{P}_s : \boldsymbol{u}_s^* = \operatorname*{argmin}_{\boldsymbol{u}_s \in \mathcal{U}_s} J_s^{\text{NMPC}}(\boldsymbol{u}_s, \xi_s) \tag{4.1}$$

where J_s^{NMPC} is the NMPC cost and $\mathcal{U}_s$ is the admissible set of control profiles $\boldsymbol{u}_s$. The vector ξ_s encapsulates all parameters such as the estimated state $\hat{x}_s$, the set-point r_s and the incoming coupling profile $\boldsymbol{v}_s^{in}$. These variables are considered frozen during the resolution of (4.1) and will be dropped in this section for a sake of compactness.

The implementation of the fast gradient method requires the gradient of the cost function at J_s^{NMPC} with respect to $\boldsymbol{u}_s$, which can be easily obtained by modeling the cost with CasADi and then computing its gradient $\nabla J_s^{\text{NMPC}} = \partial J_s^{\text{NMPC}} / \partial u_s$ by using the subroutine **jacobian**(J_s^{NMPC},$\boldsymbol{u}_s$). The algorithm that is used to solve (4.1) is given by the following updating rule:

$$\boldsymbol{z}_s^{(i+1)} = \boldsymbol{u}_s^{(i)} - \gamma^{(i+1)} \cdot \nabla J_s^{\text{NMPC}}(\boldsymbol{u}_s^{(i)}) \tag{4.2}$$

$$\boldsymbol{u}_s^{(i+1)} = \mathbf{Pr}(\boldsymbol{z}_s^{(i+1)} + c \cdot (\boldsymbol{z}_s^{(i+1)} - \boldsymbol{z}_s^{(i)}), \mathcal{U}_s) \tag{4.3}$$

where $\boldsymbol{z}_s^{(i+1)}$ is the updated vector variable and $c \in (0, 1)$ is the design variable. $\mathbf{Pr}(p, \mathcal{U}_s)$ is the projection of vector p on the admissible set $\mathcal{U}_s$. The variable γ is the adaptive step size that is computed by using Barzilai-Borwein formula proposed in [59]:

$$\gamma^{(i+1)} = \frac{\|(\boldsymbol{u}_s^{(i+1)} - \boldsymbol{u}_s^{(i)})^T \cdot (\nabla J_s^{\text{NMPC}}(\boldsymbol{u}_s^{(i+1)}) - \nabla J_s^{\text{NMPC}}(\boldsymbol{u}_s^{(i)}))\|}{\|\nabla J_s^{\text{NMPC}}(\boldsymbol{u}_s^{(i+1)}) - \nabla J_s^{\text{NMPC}}(\boldsymbol{u}_s^{(i)})\|^2} \tag{4.4}$$

In [60], the authors showed that the convergence of the algorithm could be improved when a restart mechanism is included. More precisely, the variable $\boldsymbol{u}_s$ is restarted every n_{rst} iteration, but it is noted that the frequency of restarts should depend on the cost function. This mechanism is expressed as below:

```
$\boldsymbol{z}_s^{(i+1)} = \boldsymbol{u}_s^{(i)} - \gamma^{(i+1)} \cdot \nabla J_s^{\mathrm{NMPC}}(\boldsymbol{u}_s^{(i)})$;
**if** $\mathrm{mod}(i, n_{rstr}) == 0$ **then**
    $\boldsymbol{u}_s = \mathrm{Pr}(\boldsymbol{z}^{(i+1)}, \mathcal{U}_s)$;
**else**
    $\boldsymbol{u}_s = \mathrm{Pr}(\boldsymbol{z}_s^{(i+1)} + c \cdot (z_s^{(i+1)} - z_s^{(i)}), \mathcal{U}_s)$;
**end**
```

Finally, this method is summarized by Algorithm 3. The algorithm is stopped when the number of iterations reaches $N_{\max}$

Algorithm 3 Truncated gradient-based solver

```
**Initialize:**
    $i \leftarrow 0$; $c \in (0,1)$; $\gamma^{(i)} \in (0,1)$; $n_{rstr} \in \mathbb{N}$
    $\boldsymbol{u}_s^{(i)} \leftarrow \mathbf{0}$; $\boldsymbol{z}_s^{(i)} \leftarrow \mathbf{0}$

**for** $i \leftarrow 1, \ldots, N_{\max}$ **do**
    $\boldsymbol{z}_s^{(i+1)} = \boldsymbol{u}_s^{i} - \gamma^{(i)} \cdot \nabla J_s^{loc}(\boldsymbol{u}_s^{(i)})$;
    **if** $\mathrm{mod}(i, n_{rstr}) == 0$ **then**                ▷ check for restart
        $\boldsymbol{u}_s^{(i+1)} = \mathbf{Pr}(\boldsymbol{z}_s^{(i+1)}, \mathcal{U}_s)$;
    **else**
        $\boldsymbol{u}_s^{(i+1)} = \mathbf{Pr}(\boldsymbol{z}_s^{(i+1)} + c \cdot (\boldsymbol{z}_s^{(i+1)} - \boldsymbol{z}_s^{(i)}), \mathcal{U}_s)$;
    **end if**
    Compute $\gamma^{(i+1)}$ by (4.4);
**end for**
```

4.3 Neural-network-based NMPC

In this section, the objective is to derive a regression model that predicts the values of $\boldsymbol{u}_s^*$ by basing on a learning data set in which the algorithm 3 is involed. The central idea here is to replace the implicitly defined control profile (4.1) by an explicit representation of the form $\boldsymbol{u}_s^* = K_s(\xi_s, \theta_s^*)$, where θ_s^* is the parameters that minimize the objective function given below:

$$\theta_s^* = \underset{\theta_s}{\mathrm{argmin}} \frac{1}{N_s} \sum_{i=1}^{N_s} \| u_s^{*,(i)} - K_s(\xi_s^{(i)}, \theta_s) \|^2 \tag{4.5}$$

where $\{(\xi_s^{(1)}, u_s^{*,(1)}), \ldots, (\xi_s^{(N_s)}, u_s^{*,(N_s)})\}$ is the set of N_s training data. In this section, only one subsystem is considered, the subscript s is thus omitted for the sake of simplicity.

Table 4.1: Comparison of different machine learning algorithms with the preferred properties being highlighted.

Algorithm	Function	Multivariate	Dimensions
GLM	Linear	**Yes**	**High**
VAR	Linear	**Yes**	**High**
RF	PW constant	**Yes**	**High**
SVM	**Nonlinear**	No	Moderate
NLR	**Nonlinear**	No	**High**
DNN	**Nonlinear**	**Yes**	**High**

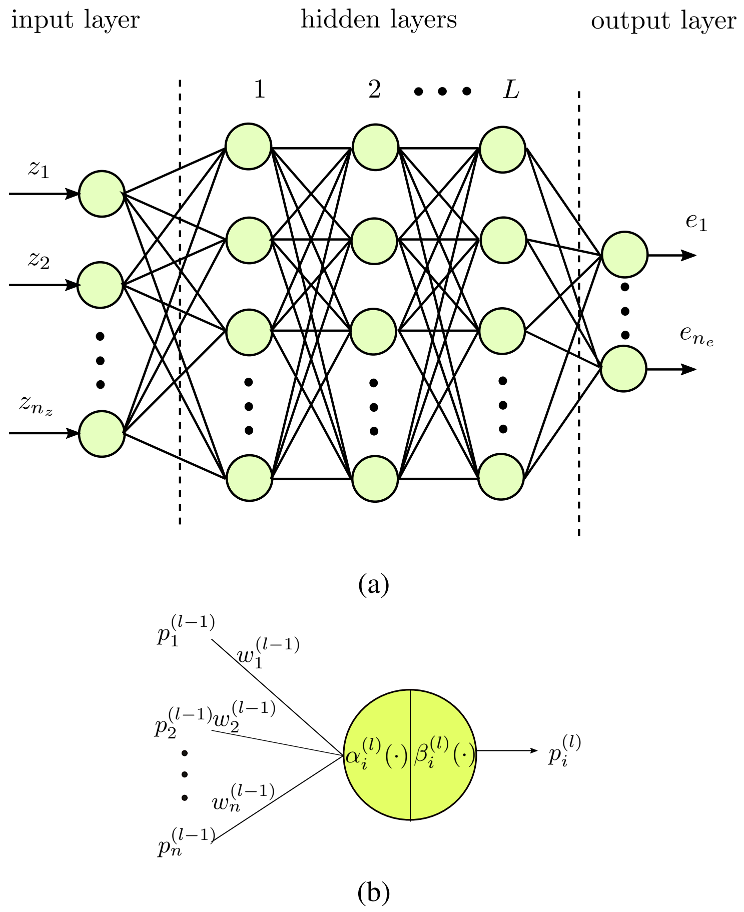
Among possible approaches, machine learning-based approach have become a popular choice to approximate the functional form $K_{ML}(\xi, \theta)$ because of their universal approximation property. In the context of approximating MPC under constraints, [61] has listed a bunch of machine learning algorithms along with their characteristics, which are summarized in Table. 4.1. The algorithms that are compared are: generalized linear models (GLM), nonlinear regression (NLR), random forest (RF), support vector machines (SVM), deep neural networks (DNN) and vector autoregressive model (VAR). The properties that are used in the comparison are: the nature of the regressor function (e.g linear, nonlinear, piecewise constant); the multivariable target handling ability; and finally the ability to handle high dimension learning dataset with many samples. Since the NMPC problem formulations involve nonlinearities in the objective function and in the constraints, it is required to choose the model with nonlinear nature for the sake of versatility and flexibility. Moreover, the models that can handle multivariate target and high dimensional data are required since the whole control profiles $\boldsymbol{u}_s^*$ are to-be predicted with the parameter vectors ξ_s considerably large. After considering all these aspect, DNNs appear to be the appropriate choice as regression model. Furthermore, DNNs could be easily implemented in any programmable logic controllers (PLCs), whose computational capabilities are not suitable for high-level solver.

A feed-forward neural network consists of several hidden layers, each layer contains many neurons. Fig. 4.1 shows a feed-forward neural network with L hidden layers and $H^{(l)}$ nodes for $l \in \{1, \dots, L\}$ per each (a) and a neural nodes (b). The i^{th} neuron in layer l takes outputs of the neurons in the preceding layer $l-1$ to produce a single binary output, namely:

$$p_i^{(l)} = \beta_i^{(l)}(\alpha_i^{(l)}) \tag{4.6}$$

$$\alpha_i^{(l)}(p^{(l-1)}) = b_i^{(l)} + \sum_{j=1}^{H^{(l-1)}} w_{ij}^{(l)} \cdot p_j^{(l-1)} \tag{4.7}$$

where w_{ij} are the weighting coefficients and $b_i^{(l)}$ is the bias of the i^{th} neuron in l^{th} layer. The function $\beta_i^{(l)}$ represents nonlinear activation function associated to the i^{th} neuron in the l^{th} layer. These functions can be rectified linear units (RELU), sigmoid, hyperbolic tangent, etc,



(a)

(b)

Figure 4.1: (a) A feed forward neural network of L hidden layers and (b) synoptic view of the i^{th} neuron in the l^{th} layer, whose output $p_i^{(l)}$ is the result of the computations of the functions $\alpha_i^{(l)}(\cdot)$ and $\beta_i^{(l)}(\cdot)$.

which are critical for ensuring the universal approximation property of neural networks.

It can be noted that before passing through the activation functions $\beta_i^{(l)}$, each hidden layer involves only affine transformation of the output of its previous layer,namely:

$$\alpha^{(l)}(p^{(l-1)}) = W^{(l)} \cdot p^{(l-1)} + b^{(l)} \tag{4.8}$$

in which $p^{(l-1)} \in \mathbb{R}^{H^{(l-1)}}$ for $l \in \{2, \dots, L+1\}$ and $p^{(0)} = z$. The weighting matrix $W^{(l)}$ and the bias vector $b^{(l)}$, for $l \in \{1, \dots, L\}$, gather all weighting coefficients and the biases related to the l^{th} layer, namely:

$$W_{ij}^{(l)} = w_{ij}^{(l)} \quad \text{for } i \in \{1, \dots, H^{(l)}\}, \text{ and } j \in \{1, \dots, H^{(l-1)}\} \tag{4.9}$$

$$b^{(l)} = [b_1^{(l),T}, b_2^{(l),T}, \dots, b_{H^{(l)}}^{(l),T}]^T \tag{4.10}$$

Then, the deep neural with L layers can be defined as an parametrized function given below:

$$K_{NN}(z;\theta) = \alpha^{(L+1)} \circ \beta^{(L)} \circ \alpha^{(L)} \circ \cdots \circ \beta^{(1)} \circ \alpha^{(1)}(z) \tag{4.11}$$

$$\beta^{(l)}(\cdot) = [\beta_1^{(l)}(\cdot), \cdots, \beta_{H^{(l)}}^{(l)}(\cdot)]^T \tag{4.12}$$

where the parameter vector $\theta = \{W^{(1)}, b^{(1)}, \ldots, W^{(L+1)}, b^{(L+1)}\}$ gathers all weights $W^{(l)}$ and biases $b^{(l)}$ in the network with appropriate dimension.

Assuming that the data $\mathcal{D} := \{(\xi^{(i)}, \boldsymbol{u}^{*,(i)})\}_{i=1}^N$, with N being the number of observations in the data, are available, any parametrized $K_{NN}(\xi, \theta)$ can be trained by minimizing some loss function such as the mean squared error:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{u}^{*,(i)} - K_{NN}(\xi^{(i)}, \theta)\|^2 \tag{4.13}$$

Once the network architecture is trained, the approximate DNN-based NMPC law $K_{NN}(z, \theta^*)$ can be used online to cheaply evaluate the optimal control input.

4.3.1 Data generation

There are two common data-generation strategies, namely open-loop and closed-loop. In open-loop data generation, the set $\mathcal{Z} \subset \mathcal{X} \times \mathcal{V}^{in} \times \mathcal{R} \times \mathcal{W}$ of possible states, incoming coupling profiles, disturbances and set-points could be created and the corresponding control profile $\boldsymbol{u}$ computed that will be added together to establish a set of data $\mathcal{D} = \{(x^{(i)}, \boldsymbol{v}^{in,(i)}, r^{(i)}, \boldsymbol{w}^{(i)}, \boldsymbol{u}^{(i)})\}_{i=1}^N$. Although very simple, this strategy can result in non physically realistic instances being included in the training data. Closed-loop strategy, on the contrary, gathers data while running a closed-loop simulation under randomly drawn physically meaningful initial states. Indeed, the majority of large-scale cryogenic systems operate under a relatively small number of regimes or operating scenarios. Each operational scenario is characterized by a few regulated and/or constrained outputs and a few large magnitude disturbances that may frequently change, while the set-points are kept unchanged for a long period of time. Hence, we propose the following data generation procedure that performs off-line simulation using the system model under the control law to collect the operationally relevant training set $\mathcal{D}$:

1. Determine the operational range of the set-points denoted by $[\underline{r}, \overline{r}]$ and the realistic range of the disturbances denoted by $[\underline{w}, \overline{w}]$:
2. Create pseudo-random binary signals (PRBS) of r in its operational ranges. In order to capture the controller's setpoint tracking behavior, the amplitude of the signal must not vary for a sufficient period of time denoted by Δt. The shape of the disturbance signal w can be chosen according to its realistic behavior. In our application, the shape of the signal w and the time period Δt will be specified in Sect. 4.4.3.1.

3. Run the closed-loop simulations that implement the above discussed hierarchical design at some chosen initial states with the created PRBS signals. Note that the data is collected during the fixed-point iterations in order to capture the relationship between the control profile $\boldsymbol{u}$ and the triplet $(r, x, \boldsymbol{v}^{in})$.

The network is trained to minimize the mean squared error criteria below:

$$J_{NN}(\theta) = \frac{1}{2}\sum_{i=1}^{N_{tr}} \|\boldsymbol{u}^{(i)} - K_{NN}(\xi^{(i)}, \theta)\|^2 \tag{4.14}$$

where $N_{tr} < N$ is the number of training observations. Indeed, Before the training process, the data set is passed through a series of data preparation techniques and finally separated into two subsets that contain N_{tr} samples and $N_{val} = N - N_{tr}$ samples, which serve to train and validate the regression model. Recall that the vector $\xi^{(j)}$ encapsulates all the parameters $x^{(i)}$, $\boldsymbol{v}^{in,(i)}$, $r^{(i)}$ and $\boldsymbol{w}^{(i)}$.

4.4 Numerical results

4.4.1 Comparison between truncated fast MPC and Casadi/IPOPT

First, we compare the control performance given by the truncated fast gradient solver presented in Sect. 4.2 and IPOPT solver of Casadi. The 4-subsystem-decomposition described in Sect. 3.5.1.1 is reused to conduct the simulation presented in this chapter. In addition, the local controllers for the Joule-Thomson cycle (S_1) and the turbine T_1 (S_4) are nonlinear MPCs.

The performance of the Ipopt (CasADi) solver and the truncated gradient solver used to solve the local optimal control problems of S_1 and S_4 are compared together. This can be done by evaluating the open-loop performance indicated by $J_s^{\text{NMPC}}(\boldsymbol{u}_s^*)$, where $\boldsymbol{u}_s^*$ is the solution of (4.1). The evaluation process is described below:

1. Create realistic set of state x_s, set-point r_s and $\boldsymbol{v}_s^{\text{in}}$ denoted by $\mathcal{D}^{\text{solver}} := \{(x_s^{(i)}, r_s^{(i)}, \boldsymbol{v}_s^{(i)})\}_{i=1}^{N_{dta}}$. Since the initial state vectors and coupling profiles are hard to be created, the data set can be obtained by following the procedure described in 4.3.1.

2. Solve the problem (4.1) by using solver Ipopt and truncated gradient at triplets $(x_s^{(i)}, r_s^{(i)}, \boldsymbol{v}_s^{(i)})$ (for $i = 1, \dots, N_{dta}$). For the configuration, several choices of maximum number of iteration $N_{\max}$ and tolerance error ϵ_{tol} of the terminal criteria for the two solvers are chosen in order to analize their effects on the computation time and the optimization performance;

3. The open-loop performances $J_s^{\text{NMPC},(i)}(\boldsymbol{u}_s^{*,\text{Ipopt}})$ and $J_s^{\text{NMPC},(i)}(\boldsymbol{u}_s^{*,\text{grd}})$ of the solver Ipopt and truncated gradient are computed. Then, the average of performance ratio $\overline{J}$ between

the two solvers is deduced, namely:

$$\overline{J} = \frac{1}{N_{dta}} \sum_{i=1}^{N_{dta}} \frac{J_s^{\text{NMPC},(i)}(\boldsymbol{u}_s^{*,\text{solver}})}{J_s^{\text{NMPC},(i)}(\boldsymbol{u}_s^{*,\text{base}})} \times 100\% \quad \text{with solver} := \{\text{grd}, \text{Ipopt}\} \tag{4.15}$$

The performance in terms of optimization and computation time is analyzed. Table 4.2 shows the maximum computation time and the open-loop performance $\overline{J}$ for several configuration of ϵ_{tol} and $N_{\max}$. For the computation of $\overline{J}$, the local costs $J_s^{\text{NMPC},(i)}(\boldsymbol{u}_s^{*,\text{base}})$ appearing at the denominator in (4.15) is chosen to be the ones resulted by using the IPOPT solver with the configuration of $\epsilon_{\text{tol}} = 10^{-4}$ and $N_{\max} = 10$. The computation time resulted by using the truncated gradient-based solver is significantly reduced, while the optimization performance is not drastically effected. This can be realized by the fact that the computation time given by the choice of $\epsilon = 10^{-4}$ and $N_{\max} = 10$ is reduced from 4.76s to 0.0499s by using gradient-based solver with $N_{\max} = 100$, whereas the performance index $\overline{J}$ is not too much changed.

Table 4.2: Performance index of Ipopt solver and truncate gradient based solver

	NMPC of S_1				NMPC of S_4			
Solver	$N_{\max}$	ϵ_{tol}	$\overline{J}[\%]$	$t_{\text{cpt}}^{\max}$ [s]	$N_{\max}$	ϵ_{tol}	$\overline{J}[\%]$	$t_{\text{cpt}}^{\max}$ [s]
Truncated gradient descend	100	_	100.2379	0.0499	100	_	101.3418	0.011
	50	_	100.2395	0.0398	50	_	101.3419	0.008
	30	_	100.2357	0.0322	30	_	101.3418	0.0043
	10	_	101.29	0.0246	10	_	101.3418	0.0014
Ipopt/Casadi	5	10^{-1}	99.999	2.746	5	10^{-1}	100	0.0589
	10	10^{-1}	100.002	3.756	10	10^{-1}	100	0.0873
	10	10^{-4}	100	4.76	10	10^{-4}	100	0.1720

Indeed, the computation time can be reduced by parametrizing the optimization vector so that its degree of freedom is reduced. Table 4.3 shows the maximum computation time and the open-loop performance $\overline{J}$ associated to the parametrization $id = [1\ 5\ 10\ 30\ 50\ 100]$ (see Sect. 2.3.2.2) for several configurations of ϵ_{tol} and $N_{\max}$. Although the degree of freedom is reduced to 6 with this parametrization, the computation time of the solver Ipopt (for $\epsilon_{\text{tol}} = 5$ and $N_{\max} = 5$) is still impractical to be integrated in our framework.

Furthermore, the real computation time with respect to the allowable maximum iteration $N_{\max}$ when using fast-NMPC solver in this hierarchical control framework should be analyzed. Fig. 4.2 shows the evolution of the computation time t_1^{final} required by S_1 to compute the optimal setpoint r^{opt} and the associated control profile $\boldsymbol{u}_1$. It can be noted that the computation time is reduced with the decrease of $N_{\max}$ until $N_{\max}$ is lower than 10. The increase in computation time as $N_{\max}$ decreases from 20 can be explained by the fact that the performance of the solver is significantly deteriorated, which prevents the convergence of the fixed-point iterations. For the simulation, $N_{\max} = 30$ is chosen for both NMPCs of S_1 and S_4.

Table 4.3: Performance index of Ipopt solver and truncate gradient based solver with parametrization id = [1 5 10 30 50 100]

	NMPC of S_1				NMPC of S_4			
Solver	$N_{\max}$	ϵ_{tol}	$\overline{J}[\%]$	$t_{\text{cpt}}^{\max}[\text{s}]$	$N_{\max}$	ϵ_{tol}	$\overline{J}[\%]$	$t_{\text{cpt}}^{\max}[\text{s}]$
Truncated gradient descend	100	_	100.4967	0.0328	100	_	100	0.0117
	50	_	100.649	0.0234	50	_	100	0.005
	30	_	100.655	0.0149	30	_	100	0.004
	10	_	114.349	0.0068	10	_	100	0.002
Ipopt/Casadi	5	10^{-1}	100.166	0.691	5	10^{-1}	100	0.0783
	10	10^{-1}	100.036	0.987	10	10^{-1}	100	0.0868
	10	10^{-4}	100	1.499	10	10^{-4}	100	0.2002

4.4.2 Benefit of using nonlinear MPC for controlling the J-T cycle

Although the benefit in terms of control performance when employing nonlinear MPC in our proposed framework has been demonstrated in Chap. 3 with the use of the NMPC at the Brayton cycle or the turbine T$_1$, we would like also to compare the control performance associated to the two configurations under the hierarchical control coordination that are: the configuration where two NMPCs for S_1 (J-T cycle) and S_4(turbine T$_1$) are implemented, and the configuration where the MPC for S_1 and the NMPC for S_4 are employed. The truncated gradient-based solver are used to solve the optimal control problem of the NMPCs in both configurations.

Fig. 4.3 shows the systems behaviors of two mentioned configurations for the scenarios where the charge $\text{NCR}_{22}^{(w)}$ increases to 650 W, which overpass the nominal power of the refrigerator. It can be seen that by employing the NMPC at S_1 the constraint on the flow rate M_{out} is more satisfied. Furthermore, by using the truncated gradient-based solver, the hierarchical control algorithm is feasible since the computation time of each subsystems is smaller than the updating period $\tau_u = 5\ s$ (Fig. 4.4).

4.4.3 Approximate NMPC by neural network

4.4.3.1 Data preparation and learning assessment

In this subsection, the most time-consuming NMPC, which is the one of the J-T cycle (S_1), will be approximated by a deep neural network. The approach described in Sect. 4.3 is proceeded by beginning with the data preparation step. Then, many deep neural network structures are used to approximate the control law based on the collected data.

There are two types of scenarios that the NMPC of the J-T cycle can meet, which are

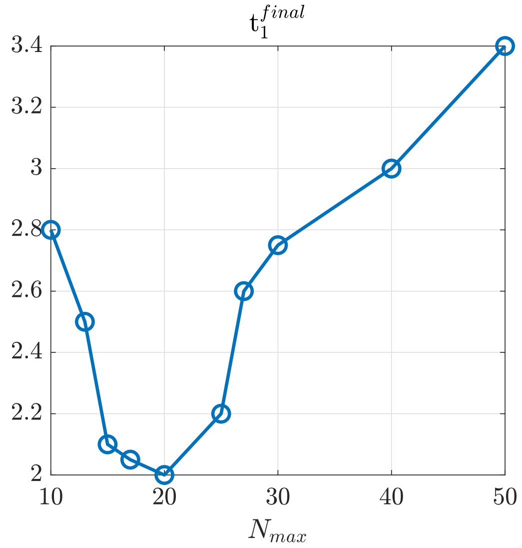


Figure 4.2: Evolution of computation time of S_1 needed for computing the optimal set-point r^{opt} and the associated control profile $\boldsymbol{u}_1$.

set-point tracking on helium level Ltb_{131} and the disturbance rejection on $\text{NCR}_{22}^{(w)}$. The PRBS profile of set-point Ltb_{131}^{sp} is created so that period Δt is long enough for the outputs to be stabilized, in our case Δt is chosen to be 1500s (Fig.4.5a). For the disturbance rejection scenario, the profile of disturbance $\text{NCR}_{22}^{(w)}$ is created so that the disturbance has the form as shown in Fig. 4.5b. The values $\overline{\text{NCR}}_{22}^{(w)}$ and $\underline{\text{NCR}}_{22}^{(w)}$ are randomly generated from period to period and in the operation range. Indeed, the disturbance can be assumed to have a specific form since it depend on operation modes of tokamaks as shown in Sect. 1.2.6.

Then, the system is simulated in closed-loop with the created profiles of r_1 and w_1, separately, under the hierarchical control algorithm described in Chap. 3. The tuples $\{(x_1^{(i)}, \boldsymbol{v}_1^{in,(i)}, r_1^{(i)}, \boldsymbol{w}_1^{(i)}, \boldsymbol{u}_1^{(i)})\}_{i=1}^{N_s}$ are collected in fixed-point iterations in order to capture all the relation between the control profiles and the relating parameters.

After the data are gathered, the data-preprocessing techniques are proceeded:

1. Data balancing: one of the important rules in machine learning is to balance out the data set. In our case, there are only two scenarios which are set-point tracking and disturbance rejection, thus, the data can be balanced by taking the same number of observations of each scenario into the data set.
2. Data normalization: This step normalizes the data so that all the features have smaller scale (e.g [-1,1]). This is necessary because the gradient descent (for training neural network) converges faster when the features in the observations haves similar value ranges.
3. Data Shuffling and data splitting: In order to reduce variance of the data set and to

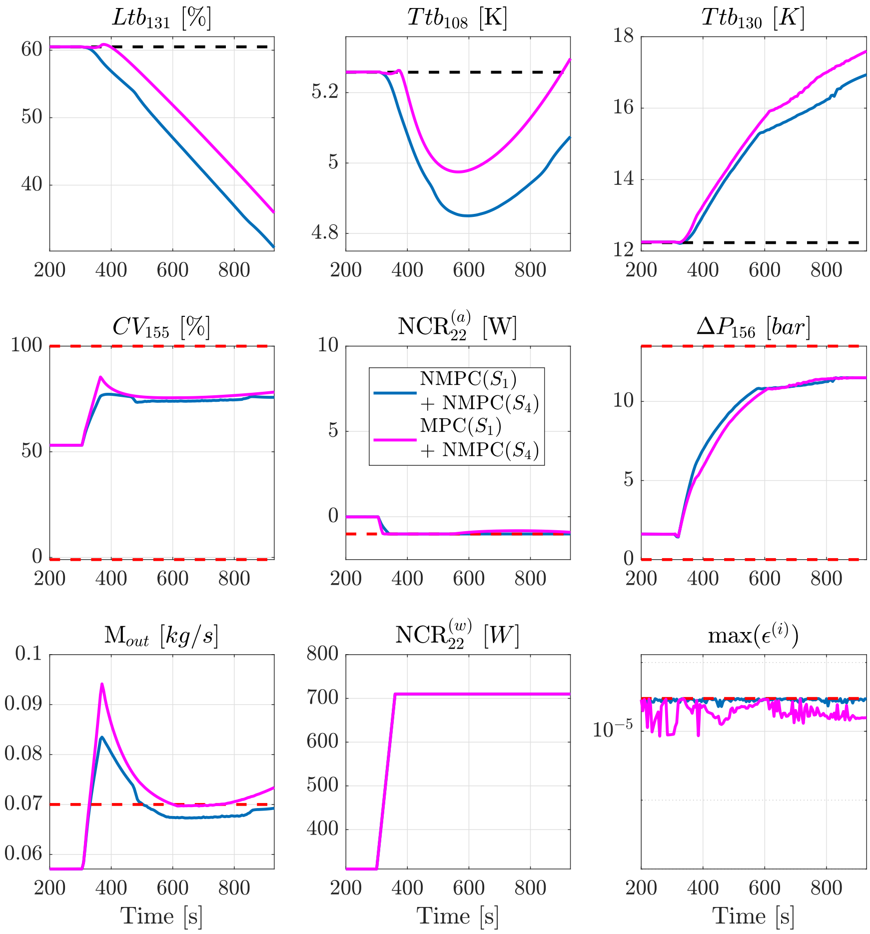


Figure 4.3: Comparison of the system behaviors given by the two configurations: using NMPC (blue) and MPC (pink) for S_1.

reduce the risk of overfitting, the collected data that are ordered temporally needs to be shuffled. The data set can be separated into n_{pack} packs, each contains n_{obs} observations (such that $n_{pack} * n_{obs} = N_s$). Then, the packs are shuffled between them. Finally, the processed data set is then split into two smaller data set which serve to train and validate the model. In our case, the data set is split such that 80% is for training data set and 20% is for validation data set.

After the data is ready, three feed-forward neural networks are trained. These configurations are set up so that each DNN has a different number of hidden layers, ranging from 1 to 3 layers, with each layer having the same number of nodes, i.e., 25 nodes, denoted by NN-1-25, NN-2-25, and NN-3-25, respectively. The activation function at each node is the sigmoid function (other activation functions have been used but do not give any better

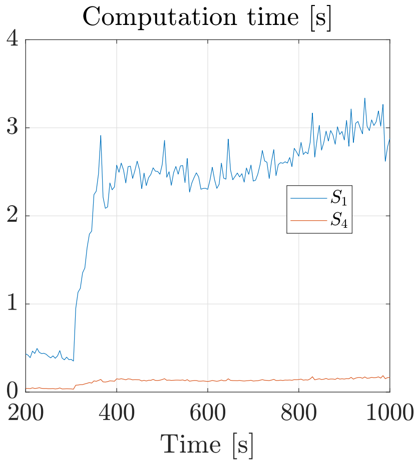


Figure 4.4: Computation time of the agents in the configuration that implement the NMPCs(S_1,S_4)

performance). Concretely, each structure is trained for 10000 epochs with the prepared data set and is validated with the validation data set. The resilient back-propagation (RPROP) algorithm is used to train the neural network. Table 4.4 presents the learning performance for three DNN structures. The structure NN-2-25, which has the lowest mean squared error (MSE) is chosen to conduct the next simulation.

Table 4.4: The learning performance of several configuration of DNNs.

Structure	NN architecture	MSE	Training time
NN-1-25	[25 25 12]	0.3192	2h47
NN-2-25	[25 25 25 12]	0.2726	3h15
NN-3-25	[25 25 23 25 12]	0.2996	3h50

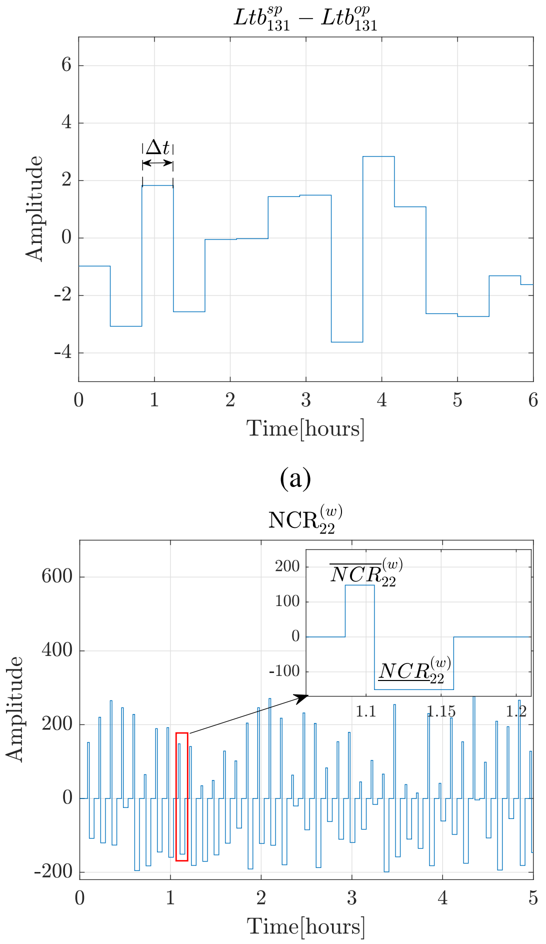
4.4.3.2 Simulation result

In order to facilitate the result interpretation, some previously defined performance indicators in the last chapter will be needed. First, the closed-loop performance indicator J_c^{CL} is recalled, namely:

$$J_{sim} = \frac{1}{N_{sim}} \sum_{s\in\mathcal{N}} \sum_{i=1}^{N_{sim}} \Big[\|y_s^{sim}(i) - r_s^d(i)\|_{Q_c^{(s)}} + \|u_s^{sim}(i)\|_{R_c^{(s)}} \tag{4.16}$$

$$+\| \max(y_s^{sim}(i) - \overline{y}_s, 0)\|_{Q_{cstr}^{(s)}} \Big] \tag{4.17}$$

where the weighting matrices $Q_c^{(s)}$, $R_c^{(s)}$ are chosen as followed:



(a)

(b)

Figure 4.5: PRBS profile of Ltb_{131}^{sp} (a) and $\mathrm{NCR}_{22}^{(w)}$ (b) that are used to generate the data for training neural networks.

- ✓ **Mode 1**: For disturbance rejecting scenario:

$$Q_c^{(1)} = \begin{bmatrix} 10^3 & 0 \\ 0 & 10^3 \end{bmatrix}, \qquad R_c^{(1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4.18}$$

$$Q_{cstr}^{(3)} = 5 \cdot 10^9 \tag{4.19}$$

$$Q_c^{(4)} = 10^3, \qquad R_c^{(4)} = 0 \tag{4.20}$$

- ✓ **Mode 2**: For set-point tracking scenario:

$$Q_c^{(1)} = \begin{bmatrix} 10^6 & 0 \\ 0 & 0.1 \end{bmatrix}, \qquad R_c^{(1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4.21}$$

$$Q_{cstr}^{(3)} = 5 \cdot 10^9 \tag{4.22}$$

$$Q_c^{(4)} = 10^4, \qquad R_c^{(4)} = 0 \tag{4.23}$$

Then, the maximum terminal error is also recalled, namely:

$$\overline{\epsilon}(k) := \max_{i=1}^{n_{ev}}(\epsilon_{end}^{(i)}) \tag{4.24}$$

where $\epsilon_{end}^{(i)}$ are the terminal convergence errors resulted from every central cost evaluation processes.

Eventually, the numerical results can be investigated. First, two configurations under the hierarchical coordination will be compared, which are the combination of two NMPCs of S_1 and S_4, and the combination of the neural-network-based NMPC of S_1 and the NMPC of S_4. Fig. 4.6 shows the system behaviors associated to these configurations in the scenarios of pulsed charge. It can be realized that the neural network can mimic the MPC actions in taking into account the coupling signals. The convergence error (subplot (3,3,9)) of two configurations are ensured to be lower than the predefined threshold $\epsilon_{\max}$.

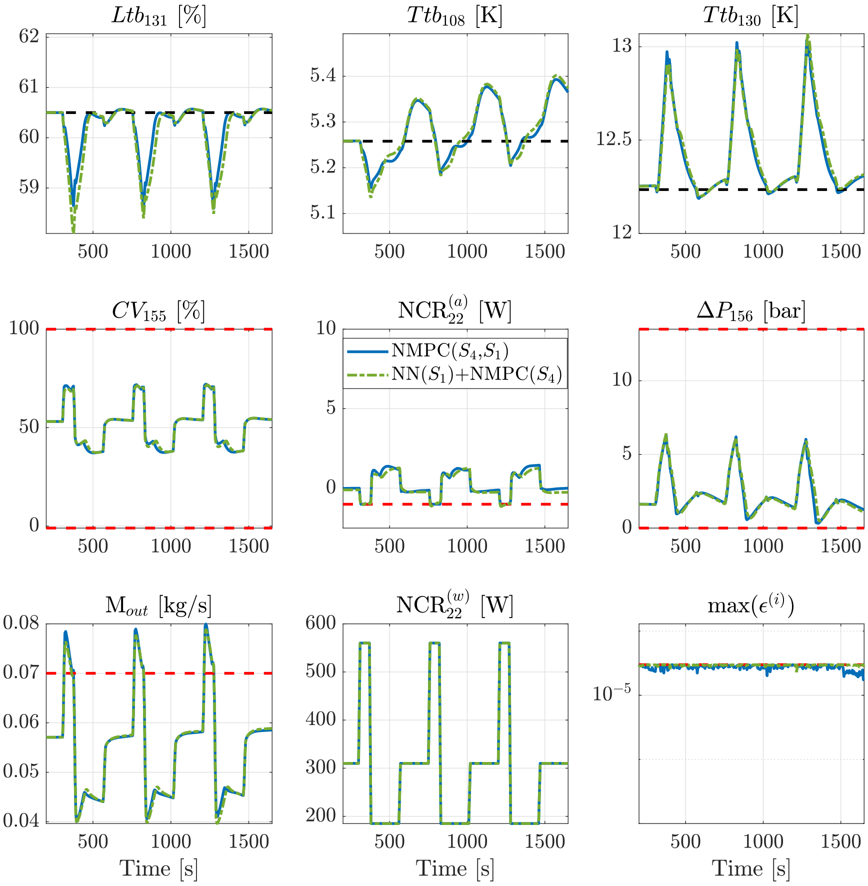


Figure 4.6: Comparison of the system behaviors when using NMPC and NN-based NMPC for S_1 in hierarchical control.

The main advantages of the machine learning controller are in the implementation burden and computational efforts. Instead of solving the optimization problem several times in the fixed-point iterations, for several set-points to be evaluated and at each sampling instants, the NN-based controller only needs to evaluate the function $\boldsymbol{u}_s^* = K(\epsilon, \theta^*)$. Consequently, the computation time resulted by the implementation of NN-based controller is reduced as shown in Fig. 4.7. It can be noted that the computation time imposed by the truncated gradient-based solver is reduced by factor 12 (the maximum computation times when using the gradient-based solver and the trained NN are 3.27 s and 0.27 s, respectively). Furthermore, this approach is applicable on typical low-level hardware, such as PLCs, since non advanced software libraries are required and the trained NN can be easily implemented with a low memory footprint. In order to take advantage of this benefit, the control input can be updated more frequently, which will improve the control performance as demonstrated in Sect. 3.5.2.3.

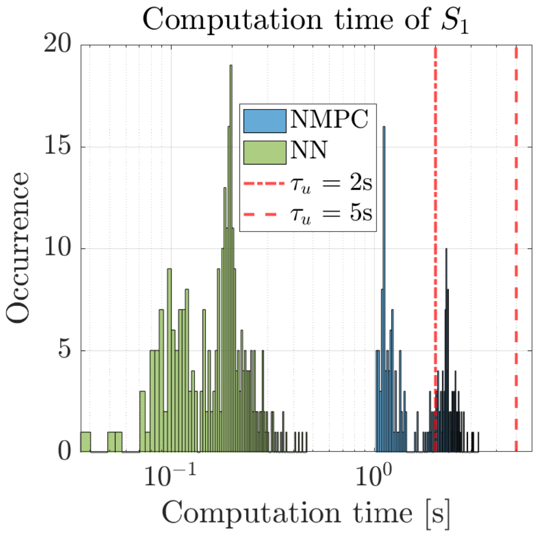


Figure 4.7: Computation time of the agents in the configuration that implement the NMPCs (S_1,S_4)

Fig. 4.8 shows the output behaviors and the closed-loop control performance associated to the previous set-up of the local controllers, under the control updating period $\tau_u = 5$ s, and the one given by using the NN-based controller at S_1 under $\tau_u = 2$ s. In the comparison between the NN-based controller and the NMPC controller with the same updating period $\tau_u = 5$ s, the cumulative performance is dropped by 18 % (at time instant t= 1600 s of subfigure (5,1)). However, this performance is recovered and even improved approximately 50 % (at t = 1600 s) when the updating period is feasibly set to be at $\tau_u = 2$ s thanks to the use of NN-based controller.

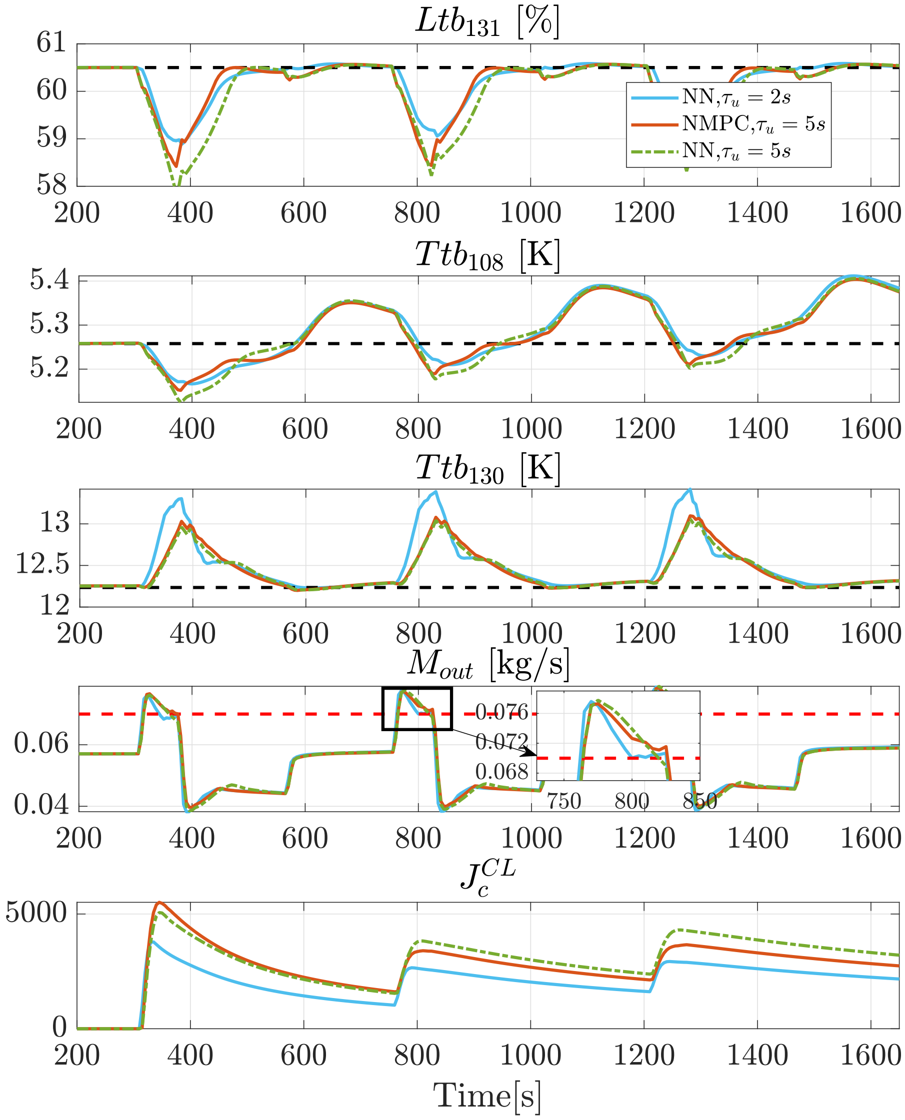


Figure 4.8: Output behaviors of the system in the case of disturbance rejecting under the coordination , in which NMPC and NN-based controller are implemented by S_1. The updating period is chosen to be $\tau_s = 5s$ and $\tau_s = 2s$ in order to compare the control performance.

Finally, the use of NN-based controller is validated in the set-point tracking scenario illustrated in Fig. 4.9. It can be seen that the system behavior under the hierarchical control method with NN-based controller and with NMPC are similar. The NN-based controller can also mimic the behavior of the NMPC of S_1 in the set-point tracking case, which results the same system behaviors with less computation efforts.

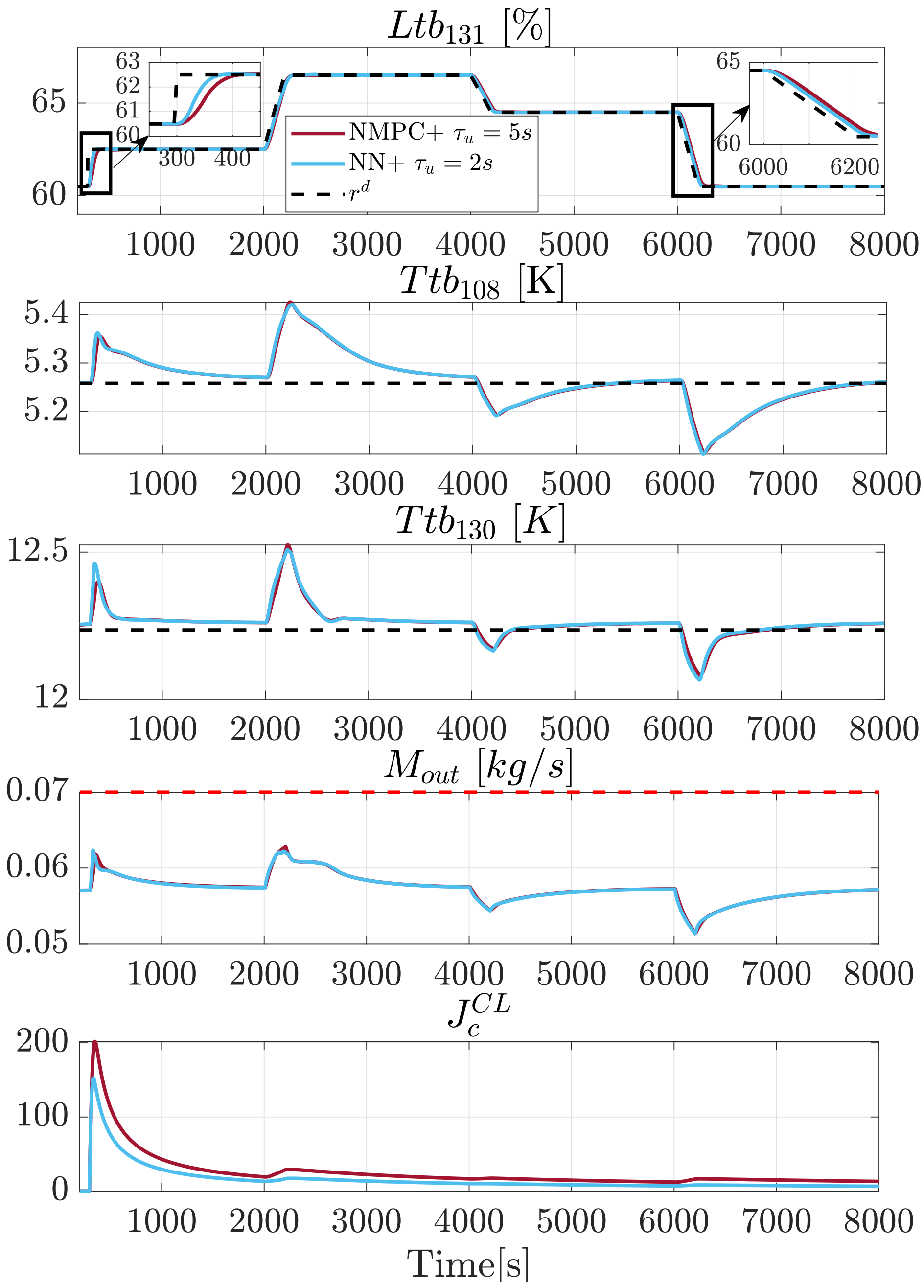


Figure 4.9: Output behaviors of the system in the case of set-point tracking under the coordination, in which NMPC and NN-based controller are implemented by S_1. The updating period is chosen to be $\tau_s = 5\ s$ and $\tau_s = 2\ s$ in order to compare the control performance.

4.5 Conclusion

In this chapter, two methods have been proposed to reduce the computation time for solving the constrained nonlinear optimization problem at the local layer of the hierarchical control framework. The numerical results have demonstrated the effectiveness of the two approaches. More precisely, the computation time is reduced drastically by using the Truncated gradient method. Then, the control law is approximated by a deep neural network. Finally, the two approaches are then compared in terms of computation time and control performance, showing that the deep learning approach successfully approximates local control laws and allows for more frequent control updates. On-going work aims to validate the control structure with a full cryogenic facility.

Chapter 5

Application to a complete refrigerator and further developments regarding the fixed-point-iteration based hierarchical control framework

Abstract The objective of this chapter is to present advances in the hierarchical control method for cryogenic processes, which is previously described in Chap. 3. These advances concern the application of this framework to a more complex scenario, where more subsystems, different time scales and different types of local controllers are considered. In addition, a residual-based iterative method is integrated to enhance the convergence of the invoked fixed-point iterations between the coordinator and the local subsystems. Facing to a more complex optimization problem in terms of optimizing vector's size, a subgradient-based solver is implemented to replace the quadratic approximation based solver (described in Chap. 3). The efficiency of the development is finally assessed through simulation-based studies.

5.1 Introduction

This chapter is an extension of the original work described in Chap. 3. While the original work has been validated by using a rather simple system (compared to real grand refrigerators), the proposed hierarchical control framework will be now applied to a more challenging control problem where the target system has more subsystems with different time scales. To achieve this goal, some contributions that need to be made are presented as follows:

- The proposed framework has been validated for both a linear case [1] and a nonlinear case under actuator constraints (see Chap. 3). Nevertheless, the same validation scheme is used, in which only two controlled cryogenic subprocesses are involved, which seems conservative. In this chapter, the framework is applied to the control of a more complex cryoplant consisting of four cryogenic subprocesses (one Joule-Thomson cycle, two Brayton cycles and a compressor station), which can be decomposed into eight subsystems. These subsystems can be either controlled or uncontrolled. Furthermore, the controlled subsystems operate on different time scales due to the difference in the their behavior characteristic times. The proposed framework is demonstrated for such an application in this chapter.

- The first development that is done in the framework concerns the method that is used to enhance the convergence of the fixed-point iteration. Recall that [1] proposed an innovative method that uses a filter to make the communication between the subsystems and the coordinator converge to a fixed-point. This method has been shown to be effective in the case where nonlinearities and actuator constraints are considered [62], if the coefficients of the local controllers are carefully tuned. However, the synthesis of this filter involves the local information that depend on the linearized models of the subsystems. This violates the modular privacy preservation requirement that was made in the original work [1]. Therefore, this chapter presents a residual-based iterative method that uses only the historical data to ensure convergence of the fixed-point iteration; thus, no local information is needed, making the entire framework directly implementable without going through the filter synthesis process.

- The second development concerns the resolution of the optimal setpoint, which is related to the central optimization problem at the coordination layer. In Chap. 3, a solver that is based on quadratic approximation is implemented to solve the global optimization problem with respect to the setpoints sent to the subsystems. This method is simple to use but its performance degrades if the dimension of the optimization vector (which is the setpoint vector) becomes larger. Indeed, the performance of this solver depends on the quality of the central cost approximation. If the complexity of the optimization problem increases, the quadratic approximation will be influenced and thus the performance of the optimization. Hence, a subgradient-based solver that is used to optimize the central cost is implemented and tested in the numerical simulation section.

This chapter is organized as follows: Sect. 5.2 introduces a 1000 W at 4.4 K refrigerator. Section 5.3 recalls the fixed-point based hierarchical control algorithm and introduces some assumptions regarding the process scheduling in this application. Section 5.5 presents some development in order to apply efficiently the method to this specific system, while Sect. 5.6 presents the simulation results.

5.2 Description of the 1000 W at 4.4 K refrigerator

This section describes the cryogenic system studied in this chapter. Figure 5.1 shows the schematic view of the 1000 W refrigerator at 4.4 K, which consists of four main cryogenic sub-processes: a Joule-Thomson cycle, a Brayton cycle, a pre-cooling Brayton cycle and a warm compression station (WCS).

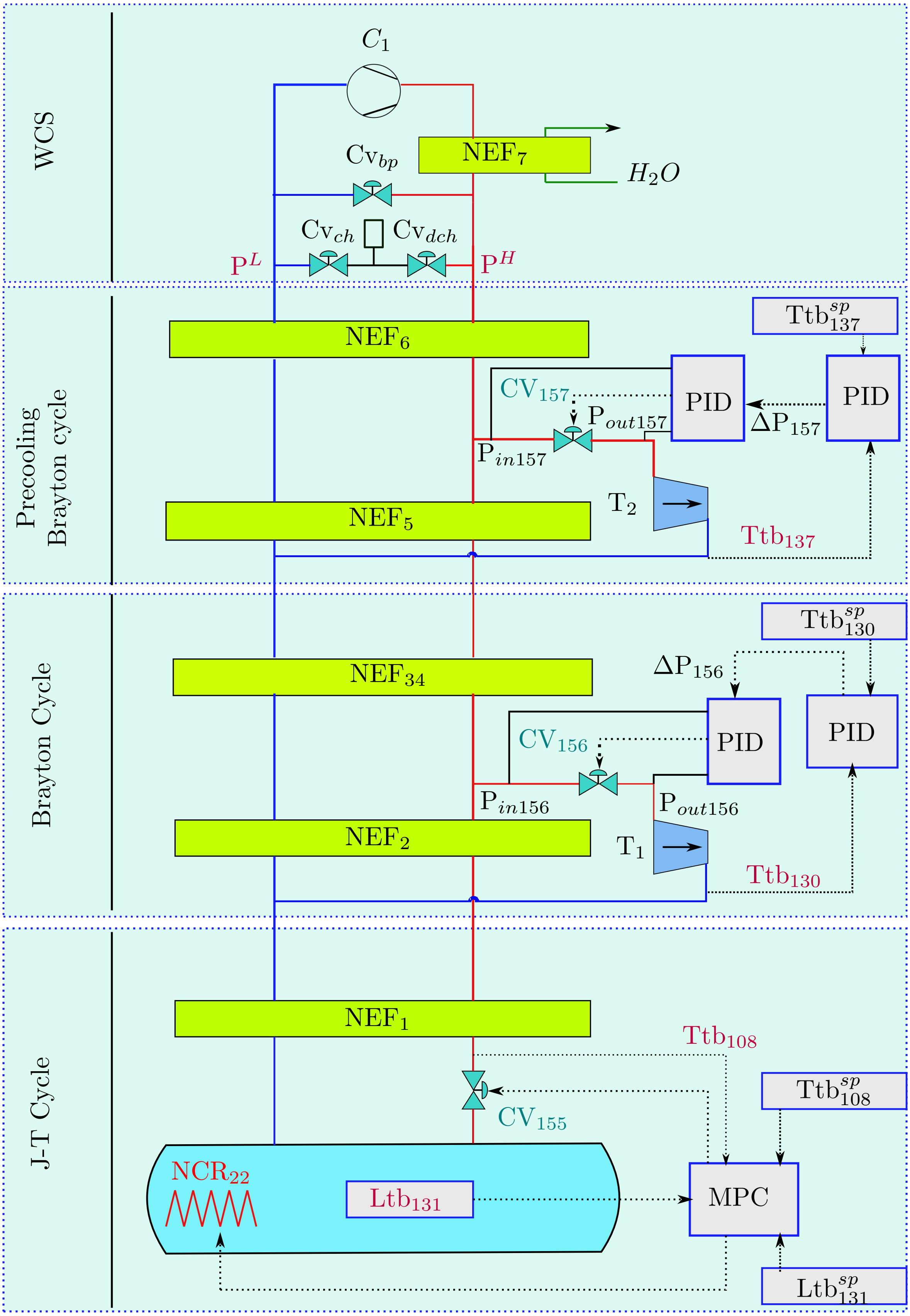


Figure 5.1: synoptic view of the cryogenic refrigerator of 1000 W at 4.4 K. Note that this refrigerator is not existing but realistic.

Briefly speaking, the gaseous helium flows clockwise in two main pipelines, which are the high-pressure pipeline (red line) and the low-pressure pipeline (blue line). The cooling power of the cryogenic refrigerator is generated by exchanging heat power in the fluid through a series of heat exchangers denoted by NEF$_x$ and also by extracting thermal energy by using two turbines denoted by T$_1$ (in the Brayton cycle) and T$_2$ (in the precooling Brayton cycle). The gaseous helium is partially liquefied by the expansion after passing through the valve CV$_{155}$ and falls into the helium bath. The low-temperature gaseous part and the evaporated part leave the bath and returns to the cycle via the low-pressure pipeline. The plant is subjected to a heating power induced by the heating source denoted by NCR$_{22}$. Finally, the cycle is closed by the so-called warm compression station (WCS), where the compressor C_1 resides. In addition, a group of valves is used to regulate the pressure at the inlet and outlet of the compressor C_1, denoted by P^L and P^H. However, it should be noted that this refrigerator is not existing but realistic since the existing refrigerators at CERN are more complex with more than 4 Brayton cycles. This refrigerator is considered as a simplified version in order to demonstrate the performance of the proposed framework.

The entire refrigerator can be decomposed into an interconnecting network of eight subsystems, as shown in Fig. 5.2. In this topology, there exist a set of subsystem indices denoted by $\mathcal{N} = \{1, ..., n_s = 8\}$, which is divided into two subsets $\mathcal{N}^{ctr}$ and $\mathcal{N}^{unc}$. The indices that belong to the subset $\mathcal{N}^{ctr}$ refer to the controlled subsystems, whereas the indices that belong to the subset $\mathcal{N}^{unc}$ refer to the uncontrolled subsystems. The controlled subsystems are the Joule-Thomson cycle, two turbines T$_1$ and T$_2$ and the WCS, while the other subsystems are uncontrolled. In addition, each subsystems S_s affects their neighbors $S_{s'}$ through the coupling signals denoted by $v_{s \to s'}$ with $s' \in \mathcal{N}_s$ (where $\mathcal{N}_s$ represents the set that contains the indices of all the neighbors of the subsystem S_s). The method that is used to identify the coupling signals of each local subsystem can be found in Sect. 2.2.

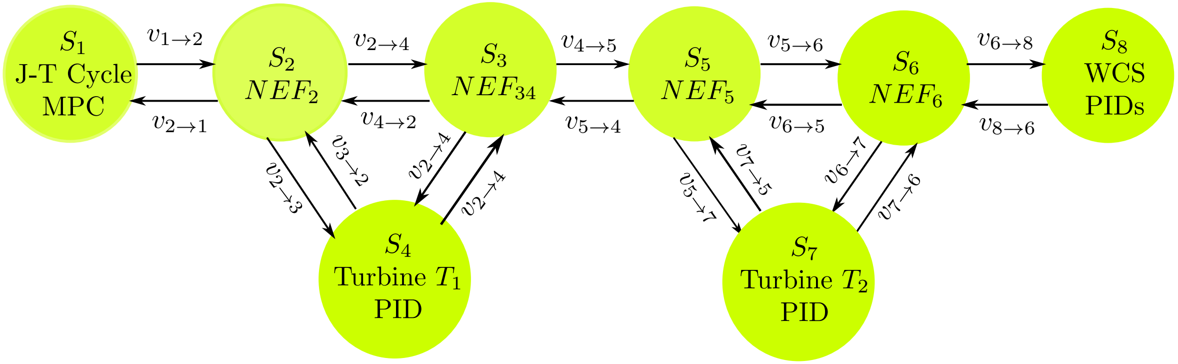


Figure 5.2: The interconnection between the subsystems of the cryogenic plant. The introduced set corresponding to this decomposition topology are $\mathcal{N} := \{1, \ldots, 8\}$; $\mathcal{N}^{ctr} := \{1, 4, 7, 8\}$; $\mathcal{N}^{unc} := \{2, 3, 5, 6\}$; $\mathcal{N}_1 := \{2\}$; $\mathcal{N}_2 := \{1, 3, 4\}$; $\mathcal{N}_3 := \{2, 4, 5\}$; $\mathcal{N}_4 := \{2, 3\}$; $\mathcal{N}_5 := \{3, 6, 7\}$; $\mathcal{N}_6 := \{5, 7, 8\}$; $\mathcal{N}_7 := \{5, 6\}$; $\mathcal{N}_8 := \{6\}$.

In order to fully understand the control context stated in this chapter, the manipulated inputs, the regulated outputs as well as the controllers implemented by the local agents are defined in the following subsections.

5.2.1 The Manipulated Inputs

There are five control inputs that are defined below (see Fig. 5.1 for the notation):

1. $\mathrm{CV}_{155} \in [0, 100]$ (%): This valve is situated at the inlet of the helium bath.
2. $\mathrm{NCR}_{22}^{(a)} \in [0, 100]$ (W): This heating actuator is located inside the helium bath (S_1). The value of $\mathrm{NCR}_{22}^{(a)}$ is in the range of $[0, 55]$ W. Note that the variable NCR_{22} in Fig. 3.4 is decomposed into two terms (as explained in Sect. 3.5.1.1):

   $$\mathrm{NCR}_{22} := \mathrm{NCR}_{22}^{(a)} + \mathrm{NCR}_{22}^{(w)} \tag{5.1}$$

   where $\mathrm{NCR}_{22}^{(w)}$ represents the disturbance coming from the heat source.
3. $\Delta\mathrm{P}_{156} \in [0, 12]$ (bar): The pressure drop between the inlet pressure and outlet pressure of the valve CV_{156}. It should be noted that the valve CV_{156} is used to control the pressure drop $\Delta\mathrm{P}_{156}$ between its inlet and outlet pressure. To do so, the first local PI controller of the turbine T_1 computes and sends an appropriate value of the pressure drop $\Delta\mathrm{P}_{156}$ to the second PI controller, which acts on the opening position of the valve CV_{156} (Fig. 5.1). This type of controller is used to hide the nonlinearity of the valve CV_{156}.
4. $\Delta\mathrm{P}_{157} \in [0, 12]$ (bar): Similarly to $\Delta\mathrm{P}_{156}$, this is the pressure drop of the valve CV_{157} and is manipulated using the same logic with two PI controllers as for $\Delta\mathrm{P}_{156}$.
5. $\mathrm{Cv}_{ch} \in [0, 100]$ (%): This valve is situated in the WCS and is used to charge helium fluid in to the circuit from the helium storage.
6. $\mathrm{Cv}_{dch} \in [0, 100]$ (%): This valve is situated in the WCS and is used to remove helium fluid from the circuit into the helium storage.
7. $\mathrm{Cv}_{bp} \in [0, 100]$ (%): This valve is situated in the WCS and is used to by-pass the helium fluid from the P^H pipeline to the P^L pipeline.

5.2.2 The Regulated Outputs

There are six regulated outputs (see Fig. 5.1 for the notation):

1. Ltb_{131} (%): The helium liquid level. The set-point for this output is chosen by the operator

2. Ttb_{108} (K): The temperature at the inlet of the J-T valve must be tightly controlled in order to ensure the efficiency of the liquefaction of the helium.
3. Ttb_{130} (K): The temperature at the turbine T_1's outlet.
4. Ttb_{137} (K): The temperature at the turbine T_1's outlet.
5. P^L (bar): The pressure at the compressor C_1's inlet in the WCS. This output is also constrained in the operational range of $[1, 1.1]$ bar.
6. P^H (bar): The pressure at the compressor C_1's outlet in the WCS. This output is also constrained in the operational range of $[12, 18]$ bar.

In the usual operation, the set-points for these regulated outputs are chosen by the operator and are presented in Table. 5.1.

Table 5.1: The operational set-points of the regulated outputs

Ltb_{131}	Ttb_{108}	Ttb_{130}	Ttb_{137}	P^L	P^H
60.5 %	4.74 K	10.69 K	30.2 K	1.05 bar	16 bar

5.2.3 The local controllers

In this configuration, the subsystems are controlled individually with different time-scales. For the J-T cycle and the turbines T_1 and T_2, their control inputs are updated every $\tau_u^{\mathrm{slow}} = 5$ s, whereas, the control action of the warm compression station, which has fast dynamic, is updated every $\tau_u^{\mathrm{fast}} = 1$ s. The choices for their local controllers are listed below:

- The J-T cycle implements a linear MPC described in Chap. 2 with the updating period $\tau_u^{\mathrm{slow}} = 5$ s.
- Turbine T_1 and T_2 implement PI controllers, while taking into account the saturation constraint on the actuators $\Delta\mathrm{P}_{156}$ and $\Delta\mathrm{P}_{157}$. Note that the nonlinear models of the turbines can also be employed in order to simulate the control profiles $\boldsymbol{u}_s$ (for $s \in \{4, 7\}$).
- The WCS is controlled by using the split-range method that has been described in Chap. 2 and will be recalled mathematically in the relating section below. This method is chosen in order to ensure offset-free tracking performance. The WCS operates with a fast control updating period $\tau_u^{\mathrm{fast}} = 1$ s compared to the other controlled subsystems, while the resolution of the optimal set-point r^{opt} takes approximately 3.5 s. During the period $[k, k+1]\tau_u^{\mathrm{slow}}$, the other subsystems are not available to cooperate with subsystem S_8 in order to compute the control action to be updated every $\tau_s^{\mathrm{fast}} = 1$ s. In order to solve this problem, an assumption on the operation of the S_8 subsystem will

be presented in Sect. 5.4. In addition, the choice of the local controller for S_8 is also crucial. The split-range method is chosen to ensure that the first control action $u_8(k)$ can be easily computed using only the output measurement and the current optimal setpoint $r_8^{\text{opt}}(k)$ without needing the coupling profile $\boldsymbol{v}_8^{\text{in}}$, while satisfying the offset-free setpoint tracking requirement.

The following subsections will describe how to compute the control profile associated to the local controllers listed above.

5.2.3.1 The MPC of Joule-Thomson cycle

The J-T cycle (S_1) is modeled by a linear state-space representation presented in Sect. 3.5.1.1; and is controlled by the linear MPC whose formulation is described previously in Sect. 2.3.2. Therefore, its control profile can be denoted by:

$$\boldsymbol{u}_1 = C_1^{\text{MPC}}(x_1, r_1, \boldsymbol{v}_1^{in}) \tag{5.2}$$

where x_1, r_1 and $\boldsymbol{v}_1^{in}$ are the states, set-point and incoming coupling profile of S_1. Finally, at every $\tau_u^{\text{slow}} = 5\ s$, the control action is updated to the subsystem.

In Chap. 2, it has been shown that the coupling profile of the JT cycle can be computed by using the initial state the control profile, namely:

$$\boldsymbol{v}_1^{out} = \mathbf{g}_1^{out}(x_1, \boldsymbol{u}_1, \boldsymbol{v}_1^{in}) \tag{5.3}$$

where $v_1^{out} = \text{col}_{s' \in \mathcal{N}_1}(v_{1 \to s'})$ and $v_1^{in} = \text{col}_{s' | 1 \in \mathcal{N}_{s'}}(v_{s' \to 1})$.

The next section presents the PI controllers to control the turbines.

5.2.3.2 The PI controllers of the turbines T$_1$ and T$_2$

The turbines are modeled by the nonlinear functions given below:

$$y_s(k) = h_s(u_s(k), v_s^{in}(k)) \text{ for } s \in \{4, 7\} \tag{5.4}$$

Having the output measurement, their control actions with respect to the PI controller law are compute as follows:

$$e_s(k) = r_s - y_s(k) \tag{5.5}$$

$$e_s^{\text{int}}(k+1) = e_s(k) + e_s^{\text{int}}(k) \tag{5.6}$$

$$u_s(k) = \mathbf{Pr}(Kp^s \cdot e_s(k) + Ki^s \cdot e_s^{\text{int}}(k+1), \mathcal{U}_s) \tag{5.7}$$

where r_s is the set-point of the outputs, while e_s and e_s^{int} are error and error integrator variables. The coefficients Kp^s and Ki^s are the controller's design parameters.

Their control profiles can be computed by updating the dynamic of the subsystems over a prediction horizon of length N with respect to the PI control updating laws. More precisely, the control action at each instant in the future (e.g $u(k), u(k+1), \dots, u_{(}k+N-1))$ is computed by performing Function 4:

Function 4 Computation of $\boldsymbol{u}_s$ (for all $s \in \{4, 7\}$)

```
Input:    $r_s, y_s(k), \boldsymbol{v}_s^{in}, e_s^{\text{int}}(k)$;
Output:   $\boldsymbol{u}_s$
for $i \leftarrow 0, \dots, N$ do

    \\Compute control action for charge valve and discharge valve
    $e_s(k+i) \leftarrow r_s - y_s(k+i)$;
    $e_s^{\text{int}}(k+i+1) \leftarrow e_s(k+i) + e_s^{\text{int}}(k+i)$;
    $u_s(k+i) \leftarrow \mathbf{Pr}(Kp^s \cdot e_s(k+i) + Ki^s \cdot e_s^{\text{int}}(k+i+1), \mathcal{U}_s)$;

    \\Simulate the system dynamic for the next iterate
    $y_s(k+i) \leftarrow h_s(u_s(k+i), v_s^{in}(k+i))$
end for
```

For the sake of brevity, the control profiles $\boldsymbol{u}_s = [u_s^T(k), \dots, u_s^T(k+N-1)]^T$ (for $s \in \{3, 4\}$) are simply defined by a function of the set-point, the output measurement, the incoming coupling profile and the error integrator, namely:

$$\boldsymbol{u}_s = C_s^{\text{PI}}(r_s, y(k), \boldsymbol{v}_s^{in}, e_s^{\text{int}}(k)) \text{ for } s \in \{4, 7\} \tag{5.8}$$

As mentioned previously, control actions of these subsystems are updated at every $\tau_u^{\text{slow}} = 5$ s.

As stated in Sect. 3.5.1.1, the coupling profile of these turbines can be computed by simulating their dynamics with the control profiles, which can be expressed briefly as follows:

$$\boldsymbol{v}_s^{out} = \mathbf{g}_s^{out}(x_s, \boldsymbol{u}_s, \boldsymbol{v}_s^{in}) \tag{5.9}$$

with $v_s^{out} = \text{col}_{s' \in \mathcal{N}_s}(v_{s \to s'})$ and $v_s^{in} = \text{col}_{s' | s \in \mathcal{N}_{s'}}(v_{s' \to s})$, for $s \in \{4, 7\}$.

The next section presents the split-range control method to control the warm compression station.

5.2.3.3 The split-range control of the Warm compression station

The warm compression station has two outputs (high pressure P^H and low pressure P^L) to be regulated by means of three actuators (Cv_{ch}, Cv_{dch} and Cv_{bp}). Comparing to other subsystems, the dynamic of the WCS is relatively fast, which requires the control action to be updated more frequently at every $\tau_u = 1\ s$.

In order for the first action in the control profile to be computed regardless the current coupling signal, the split-range method (Fig. 5.3) is specifically chosen to control this subsystem and to be integrated into the hierarchical framework. The split-range method constitutes of two PI controllers, one for each regulated output. The first PI controller regulate the low pressure P^L by the means of by-pass valve Cv_{bp}. To regulate the high pressure P^H, the principle is: the couple of valves Cv_{ch} and Cv_{dch} is considered to be one actuator, which is manipulated by the second PI controller. If the pressure P^H is too high, the discharge valve Cv_{dch} is opened to add more gas from tge stockage into the circuit, otherwise, if the pressure P^H is too low, the charge valve Cv_{ch} is opened to reject gas from the circuit. The state-space representation of the WCS is given as follows:

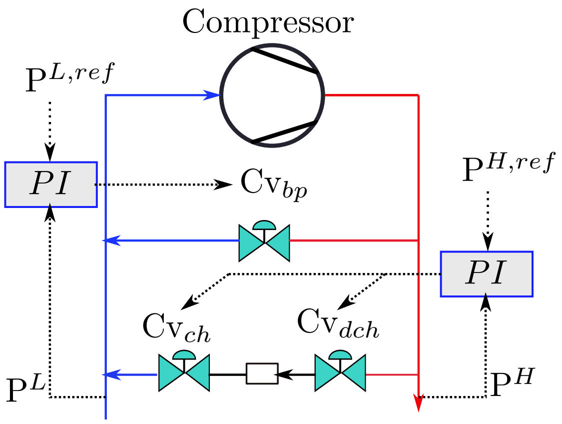


Figure 5.3: Synoptic view of split-range control method applied to the WCS.

$$x_8^+ = A_8 \cdot x_8 + B_8 \cdot u_8 + G_8 \cdot v_8^{in} \tag{5.10}$$

$$y_8 = C_8 \cdot x_8 \tag{5.11}$$

$$v_{8\to s'} = Cv_{8\to s'} \cdot x_8 \text{ for } s' \in \mathcal{N}_8 \tag{5.12}$$

where $x_8 \in \mathbb{R}^2$ is the states vector, $u = [Cv_{ch}, Cv_{dch}, Cv_{bp}]^T \in \mathcal{U}_8$ is the control input vector and $v_8^{in} \in \mathbb{R}^3$ is the incoming coupling signal. According to the previously mentioned control principle, the manipulated inputs can be computed by the following equations:

- For controlling the high pressure P^H:

$$e_{PH}(k) = r_{PH} - P^H(k) \tag{5.13}$$

$$e^{\text{int}}_{PH}(k+1) = e_{PH}(k) + e^{\text{int}}_{PH}(k) \tag{5.14}$$

$$u_{PH}(k) = Kp^{8,P^H} \cdot e_{PH}(k) + Ki^{8,P^H} \cdot e^{\text{int}}_{PH}(k+1) \tag{5.15}$$

$$Cv_{ch}(k) = \min(\max(u_{PH}(k), 0), 100) \tag{5.16}$$

$$Cv_{dch}(k) = \min(\max(-1 \cdot u_{PH}(k), 0), 100) \tag{5.17}$$

  where r_{PH} is the set-point of the high pressure P^H. The actual error and error integrator are respectively presented by e_{PH} and e^{int}_{PH}. The equations (5.16)-(5.17) project the values on their admissible set which is [0,100] %. The coefficients Kp^{8,P^H} and Ki^{8,P^H} are the controller's design parameters.

- For controlling the low pressure P^H:

$$e_{PL}(k) = r_{PL} - P^L(k) \tag{5.18}$$

$$e^{\text{int}}_{PL}(k+1) = e_{PL}(k) + e^{\text{int}}_{PL}(k) \tag{5.19}$$

$$u_{PL}(k) = Kp^{8,P^L} \cdot e_{PL}(k) + Ki^{8,P^L} \cdot e^{\text{int}}_{PL}(k+1) \tag{5.20}$$

$$Cv_{bp}(k) = \min(\max(u_{PL}(k), 0), 100) \tag{5.21}$$

  where r_{PL} is the set-point of the low pressure P^L. The actual error and error integrator are respectively presented by e_{PL} and e^{int}_{PL}. Similarly, the equation (5.21) projects the value on its admissible set which is [0,100] %. The coefficients Kp^{8,P^L} and Ki^{8,P^L} are the controller's design parameters.

Note that the model expressed by (5.10)-(5.12) is obtained by discretizing the continuous-time model with the time constant $\tau_u^{\text{slow}} = 1\ s$. Thus, the horizon length $N_{\text{slow}} \cdot \tau_u^{\text{slow}}$ of the prediction profiles of the subsystem S_8 does not necessarily have the same length as the one of the slow subsystems (S_1, S_4 and S_7), which is $N \cdot \tau_u^{\text{fast}}$. In order to have the same horizon length between the subsystems, the length of the prediction profile of the WCS can be deduced as follows:

$$N_{\text{slow}} = N \cdot \frac{\tau_u^{\text{fast}}}{\tau_u^{\text{slow}}} = 5 \cdot N \tag{5.22}$$

It can be seen that the control profile of S_8 does not have the same length as the control profiles of S_1, S_4 and S_7, that is:

$$\boldsymbol{u}_8 = [u_8(k)^T, u_8(k+1)^T, u_8(k+2)^T, \dots, u_8(k+N_{\text{slow}}-1)^T]^T \tag{5.23}$$

Given the set-point vector $r_8 = [r_{PH}, r_{PL}]^T$, the current output measurement $y_8 = [P^H(k), P^L(k)]^T$ of the high pressure and low pressure and the incoming coupling profile $\boldsymbol{v}_8^{in}$, the control profile $\boldsymbol{u}_8$ associated to the split-range method defined over a prediction horizon of length N_{slow} can be computed by simulating the subsystem dynamic with respect to the split-range control method as shown in Function 5. Similarly, the control profile can be presented briefly by:

$$\boldsymbol{u}_8^{\text{slow}} = C_8^{\text{split-range}}(r_8, y_8(k), \boldsymbol{v}_8^{in}, e_8^{\text{int}}(k)) \tag{5.24}$$

Function 5 Computation of $\boldsymbol{u}_8$

```
Input:   $r_8 = [r_{PH}, r_{PL}]$, $y_8(k) = [P^H(k), P^L(k)]^T$, $\boldsymbol{v}_8^{in}$, $e_8^{\text{int}}(k)$;
Output:   $\boldsymbol{u}_8$
for $i \leftarrow 0, \dots, N_{\text{slow}}$ do

    \\Compute control action for charge valve and discharge valve
    $e_{PH}(k+i) \leftarrow r_{PH} - P^H(k+i)$;
    $e_{PH}^{\text{int}}(k+i+1) \leftarrow e_{PH}(k+i) + e_{PH}^{\text{int}}(k+i)$;
    $u_{PH}(k+i) \leftarrow Kp \cdot e_{PH}(k+i) + Ki \cdot e_{PH}^{\text{int}}(k+i+1)$;
    $Cv_{ch}(k+i) \leftarrow \max(u_{PH}(k+i), 0)$;
    $Cv_{dch}(k+i) \leftarrow \max(-1 \cdot u_{PH}(k+i), 0)$;

    \\Compute control action for by-pass valve
    $e_{PL}(k+i) \leftarrow r_{PL} - P^L(k+i)$;
    $e_{PL}^{\text{int}}(k+i+1) \leftarrow e_{PL}(k+i) + e_{PL}^{\text{int}}(k+i)$;
    $Cv_{bp}(k+i) \leftarrow Kp \cdot e_{PL}(k+i) + Ki \cdot e_{PL}^{\text{int}}(k+i+1)$;

    \\Project on admissible set
    $\boldsymbol{u}_8(k) = \mathbf{Pr}([Cv_{ch}(k+i), Cv_{dch}(k+i), Cv_{bp}(k+i)]^T, \mathcal{U}_8)$;

    \\Simulate the system dynamic for the next iterate
    $x_8(k+i+1) \leftarrow A_8 \cdot x_8(k+i) + B_8 \cdot u_8 + G_8 \cdot v_8^{in}$;
    $y_8(k+i) \leftarrow C_8 \cdot x_8(k+i+1)$
    $P^H(k+i) \leftarrow y_{8,1}(k+i)$
    $P^L(k+i) \leftarrow y_{8,2}(k+i)$
end for
```

Note that the above control profile is the prediction over the horizon length of $N_{\text{slow}} \cdot \tau_u^{\text{slow}}$. In order to be compatible with the prediction profiles of the slow subsystems, the final control profile can be computed by:

$$\boldsymbol{u}_8 = \Pi_{\text{slow}} \cdot C_8^{\text{split-range}}(r_8, y_8(k), \boldsymbol{v}_8^{in}, e_8^{\text{int}}(k)) \tag{5.25}$$

where $\Pi_{\text{slow}} \in \mathbb{R}^{3 \cdot N \times 3 \cdot N_{\text{slow}}}$ is the matrix that selects the elements $u_8(k)$, $u_8(k+5)$, $u_8(k+10)$, $\dots$, $u_8(k + 5 \cdot (N_{\text{slow}} - 1))$.

The outcoming coupling profile of the WCS can also be deduced, namely:

$$\boldsymbol{v}_8^{out} = \Pi_{\text{slow}}^{v} \cdot \mathbf{g}_8^{out}(x_8, \boldsymbol{u}_8, \boldsymbol{v}_8^{in}) \tag{5.26}$$

Table 5.2: The manipulated inputs u_s, disturbance input w_s ,regulated outputs y_s and controller type of the controlled subsystems.

Subsystem	u_s	w_s	y_s	Controller type
S_1	$\mathrm{NCR}_{22}^{(a)}$ CV_{155}	$\mathrm{NCR}_{22}^{(w)}$	Ltb_{131} Ttb_{108}	MPC
S_4	$\Delta\mathrm{P}^{156}$	_	Ttb_{130}	PI
S_7	$\Delta\mathrm{P}^{157}$	_	Ttb_{137}	PI
S_8	Cv_{ch} Cv_{dch} Cv_{bp}	–	P^H P^L	split-range

where $\Pi^v_{\text{slow}} \in \mathbb{R}^{3\cdot N \times 3\cdot N_{\text{slow}}}$ is the matrix that selects the elements $v_8^{out}(k)$, $v_8^{out}(k+5)$, $v_8^{out}(k+10), \dots, v_8^{out}(k+5\cdot(N_{\text{slow}}-1))$.

To conclude Sect. 5.2.3, Table 5.2 summarizes the the manipulated inputs u_s, disturbance input w_s, regulated outputs y_s and controller type of the controlled subsystems.

5.3 Recall on Fixed-point-iteration based hierarchical control

Recall that there is a central problem that is solved in the coordination layer, namely:

$$r^{\text{opt}} = \underset{r}{\text{argmin}} \;\; J_c(r, \boldsymbol{v}^{in}) \tag{5.27}$$

$$\text{subject to: } \boldsymbol{v}^{in} = \boldsymbol{g}_{out}(r, \boldsymbol{v}^{in}) \tag{5.28}$$

The fixed-point-iteration based algorithm described in Chap. 3 could be separated in to two sub-processes, namely:

Estimate central cost $J_c(r, \boldsymbol{v}^{in})$: For given setpoints $r_{s,s\in\mathcal{N}_{ctr}}$ sent by the coordinator to the subsystems, the process below is launched:

1. Step 1: The coordinator sends an initial guess $\boldsymbol{v}_s^{in,(\sigma=0)}$;
2. Step 2: The subsystems $S_{s,s\in\mathcal{N}}$ evaluate the corresponding outgoing coupling profiles $\hat{\boldsymbol{v}}_s^{out,(\sigma)}$ and send them to the coordinator;
3. Step 3: The coordinator constitutes the resulting incoming coupling profile $\hat{\boldsymbol{v}}_s^{in,(\sigma)}$ and updates it into $\boldsymbol{v}_s^{in,(\sigma+1)}$ for the next iteration.

4. Step 4: The algorithm stops when the conditions on the convergence error or on the maximum iteration number, namely, $\epsilon := \max(\|\boldsymbol{v}^{in,(\sigma+1)} - \boldsymbol{v}^{in,(\sigma)}\|) \leq \epsilon_{\max}$ and $\sigma \geq \sigma_{\max}$ are reached, if not, iterate $\sigma := \sigma + 1$ and repeat from step 1.

The consensus constraint (5.28) is satisfied if the fixed-point iteration converges to a value $\boldsymbol{v}^{in,(\infty)}$. In practice, the iteration stops as soon as the termination criteria $\epsilon := \max(|\boldsymbol{v}^{in,(\sigma+1)} - \boldsymbol{v}^{in,(\sigma)}|) \leq \epsilon_{\max}$ is reached. After the convergence of fixed-point iteration, the subsystems can compute their local costs J_s and send them to the coordinator, which allow the coordinator to compute the central cost $J_c(r, \boldsymbol{v}^{in,(\infty)})$.

In Chap. 3, the fixed-point iterations convergence is enhanced by the mixing method described in Sect. 3.3.2. However, this method requires the local subsystem information in order to synthesize the matrix filter Π, which violates the predefined modular privacy preservation requirement. In Sect. 5.5.1, a residual-based iterative method is described to replace the mixing method and make the whole framework more modular.

Optimizing the central cost: In order to optimize the problem (5.27)-(5.28), any derivation-free optimization algorithm can be used to find the optimal setpoint r^{opt}, such as BOBYQA [40], the genetic algorithm [41],... In Chap. 3, an algorithm based on quadratic approximation is proposed to solve the central optimization problem in order to find the optimal set point r^{opt}. However, this algorithm is not suitable for the target problem of this chapter where more than three setpoints need to be optimized (there are six setpoints to consider) because the candidate setpoint grid for the approximation needs to be selected appropriately. In Sect. 5.5.2, an optimization solver based on the gradient descent method is introduced to replace the last solver.

As mentioned in the introduction section, the proposed framework will be applied to a more complex system where there are multiple controlled outputs and different updating control periods to be considered. The following section will present an assumption that is needed for the feasibility of this framework to operate in a two-time-scale environment, where the slow control updating period τ_u^{slow} associated to the slow behaving subsystems (e.g. the JT cycle and the Brayton cycle) and the fast one τ_u^{fast} associated to the fast behaving subsystem such as the compression station exist simultaneously.

5.4 Assumption of two-updating-period operation

In the configuration where MPC/NMPCs are exclusively implemented to control the subsystems, the entire hierarchical control algorithm need to be successfully executed within a control updating period $[k, k+1]\tau_u$ in order to guarantee the closed-loop performance. Furthermore, the control action resulted from solving the MPC problem requires the coupling profile $\boldsymbol{v}$, which involves the fixed-point iteration communication between the subsystems, making the integration of this method in our system more complex. In this application, there are two control updating periods under which the subsystems operate, which are $\tau_u^{\text{fast}} = 1$ s and $\tau_u^{\text{slow}} = 5$ s. By choosing the local controllers as described previously, i.e, the PI controllers to control the turbines and the WCS and the MPC for controlling the J-T cycle, the following assumptions is needed:

Assumption

The computation of the optimal set-point and the control actions of the subsystems follows the assumptions described below:

- The computation of the optimal set-point r^{opt} is performed at every instant $k \cdot \tau_u^{\text{slow}}$ (with $k \in \mathbb{N}$). Consequently, the control actions of the J-T cycle and the turbines with respect to their controller types and to the computed optimal set-point are updated at every $\tau_u^{\text{slow}} = 5$ s.

- The WCS is assumed to have two processors that are $S_8^{(1)}$ and $S_8^{(2)}$. These processors operate under two different control updating periods τ_u^{slow} and τ_u^{fast}. Each processor has its own task. The first processor $S_8^{(1)}$ participates in the resolution of the central problem to compute the optimal setpoint r^{opt}, whose computation is needed to be within the slow control updating period $[k, k+1]\tau_u^{\text{slow}}$. The second processor $S_8^{(2)}$ computes the control input following the PI control laws and associated to the current optimal set-point $r^{\text{opt}}(k)$. The computation of such control action is executed at every $\tau_u^{\text{fast}} = 1$ s. Note that by choosing the PI controllers as local controllers for the S_8 subsystem, the control input can be easily computed by $S_8^{(2)}$ using only the current output measurement and the setpoint without needing the coupling profile as using MPC at each instant $k \cdot \tau_u^{\text{slow}} + j \cdot \tau_u^{\text{fast}}$ ($\forall k, j \in \mathbb{N}$). Figure 5.4 illustrates the task schedule of the processors of the subsystems.

This section has presented an assumption needed for the subsystem S_8 to be integrated in the proposed control framework. The next sections will present some advances mentioned in Sect. 5.3, which concern a new method to converge the fixed-point iterations and a gradient-based optimization solver to solve the central problem.

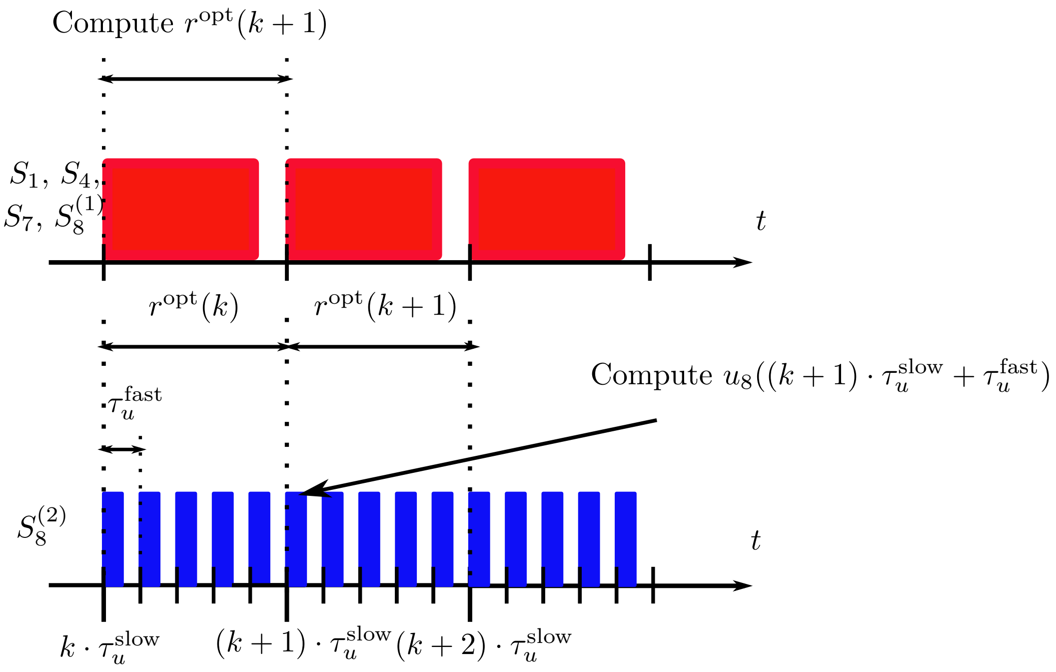


Figure 5.4: Task schedule of the processors of the subsystems.

5.5 Some development in fixed-point-iteration based control algorithm

5.5.1 Anderson method for fixed-point iteration

The Anderson method (AM) [63] is a residual-based iterative method that is used in order to enhance the convergence of any fixed-point iteration. In order to introduce the principle of the AM, let's rewrite (5.28) as a general fixed-point equation:

$$\boldsymbol{v}^{in} = G(\boldsymbol{v}^{in}) \tag{5.29}$$

AM aims to enhance the convergence of any fixed-point iteration by only using information from the most recent m_σ values $\boldsymbol{v}^{in,(\sigma)}$. More precisely, the ideal AM update at the σ-th iteration is given by:

$$\boldsymbol{v}^{in,(\sigma+1)} = G\left(\boldsymbol{v}^{in,(\sigma)}\right) - \sum_{j=1}^{m_\sigma} \gamma_j^{(\sigma)} \cdot \left[G\left(\boldsymbol{v}^{in,(\sigma-m_\sigma+j)}\right) - G\left(\boldsymbol{v}^{in,(\sigma-m_\sigma+j-1)}\right)\right] \tag{5.30}$$

The function $G(\cdot)$ is kept locally at the local subsystems and is not informed to the coordinator. The coordinator thus receives the estimates of the coupling profiles computed by the subsystems following the process described below.

First, the subsystems compute the outgoing coupling profile given the incoming coupling profile $\boldsymbol{v}^{in,(\sigma)}$, namely:

$$\hat{\boldsymbol{v}}^{out,(\sigma)} = \boldsymbol{g}_{out}\left(\boldsymbol{v}^{in,(\sigma)}\right) \tag{5.31}$$

Then, the estimation of the incoming coupling profile can be computed by the coordinator by rearranging the elements of $\hat{\boldsymbol{v}}^{out,(\sigma)}$ by using matrix G_{in}, namely:

$$\hat{\boldsymbol{v}}^{in,(\sigma)} = G_{in} \cdot \hat{\boldsymbol{v}}^{out,(\sigma)} \tag{5.32}$$

Note that by combining the equations (5.31) and (5.32) we obtain the same fixed-point equation (5.29) by defining:

$$G(\cdot) = G_{in} \cdot \boldsymbol{g}_{out}(\cdot) \tag{5.33}$$

Let us define the residual function by:

$$\begin{aligned} g_\sigma := g(\boldsymbol{v}^{in,(\sigma)}) &= G\left(\boldsymbol{v}^{in,(\sigma)}\right) - \boldsymbol{v}^{in,(\sigma)} \\ &= \hat{\boldsymbol{v}}^{in,(\sigma)} - \boldsymbol{v}^{in,(\sigma)} \end{aligned} \tag{5.34}$$

The updating rule becomes:

$$\begin{aligned} \boldsymbol{v}^{in,(\sigma+1)} =& \boldsymbol{v}^{in,(\sigma)} + g_\sigma - \sum_{j=1}^{m_\sigma} \gamma_j^{(\sigma)} \cdot \left[\left(\boldsymbol{v}^{in,(\sigma-m_\sigma+j)} - \boldsymbol{v}^{in,(\sigma-m_\sigma+j-1)} \right) \right. \\ & \left. - \left(g_{\sigma-m_\sigma+j} - g_{\sigma-m_\sigma+j-1} \right) \right] \end{aligned} \tag{5.35}$$

Let's define the matrices below:

$$\begin{aligned} \mathcal{V}_\sigma &= \left[\boldsymbol{v}^{in,(\sigma-m_\sigma+1)} - \boldsymbol{v}^{in,(\sigma-m_\sigma)} \ldots \boldsymbol{v}^{in,(\sigma)} - \boldsymbol{v}^{in,(\sigma-1)} \right] \\ \mathcal{G}_\sigma &= \left[g_{\sigma-m_\sigma+1} - g_{\sigma-m_\sigma} \cdots g_\sigma - g_{\sigma-1} \right] \end{aligned}$$

The equation (5.35) becomes:

$$\boldsymbol{v}^{in,(\sigma+1)} = \boldsymbol{v}^{in,(\sigma)} + g_\sigma - (\mathcal{V}_\sigma + \mathcal{G}_\sigma) \cdot \gamma^{(\sigma)} \tag{5.36}$$

The optimal vector $\gamma_j^{(\sigma)}$ are chosen in order to minimize the distance between $g(\boldsymbol{v}^{in,(\sigma)})$ and the linear combination of the differences $\sum_{j=1}^{m_\sigma}[g_{\sigma-m_\sigma+j} - g_{\sigma-m_\sigma+j-1}] \cdot \gamma_j^{(\sigma)}$, namely:

$$\gamma^{(\sigma)} = \underset{\gamma \in \mathbb{R}^{m_\sigma}}{\operatorname{argmin}} \, ||g_\sigma - \mathcal{G}_\sigma \cdot \gamma||^2 \tag{5.37}$$

Note that (5.37) is nothing more than a quadratic optimization problem and its solution is expressed by:

$$\gamma^{(\sigma)} = (\mathcal{G}_\sigma^T \cdot \mathcal{G}_\sigma)^{-1} \cdot \mathcal{G}_\sigma^T \cdot g_\sigma \tag{5.38}$$

Note also that periodic restarts can be included in the Anderson algorithm, meaning that the scheme is restarted periodically using only the information from the most recent iterations.

Such restarting mechanism is well known in the numerical analysis literature concerning conjugate gradient and quasi-Newton iterations to cite but few examples [64, 40].

In the following investigations, following the proposition made by [65], the original AM algorithm is modified to include systematic restarts instead of adaptive restarts. Specifically, at some iterations at the beginning of the algorithm, columns are added to the $\mathcal{V}_\sigma$ and $\mathcal{G}_\sigma$ matrices, while their allowed number of columns m_σ is incremented over iterations. Until m_σ reaches the maximum number of columns defined by m, the algorithm is restarted using only the one-column version of $\mathcal{V}_\sigma$ and $\mathcal{G}_\sigma$ in the next iteration and the matrices $\mathcal{V}_\sigma$ and $\mathcal{G}_\sigma$ continue to be filled in until they reach the maximum number of columns m. The process of building the one-column to m-column $\mathcal{V}_\sigma$, $\mathcal{G}_\sigma$ matrices can be considered a single "cycle" and after reaching the end of the cycle, this process is restarted. This modified AM scheme to include systematic restarts is detailed in Algorithm 6.

This subsection has presented an algorithm that is used to enhance the convergence of the fixed-point iteration for a given set-point sent by the coordinator. In order to find the optimal set-point, the following section will describe a gradient based method for the coordinator to iteratively find the optimal solution of the central problem expressed by (5.27)-(5.28).

5.5.2 Gradient-based solver for the coordination problem

In Chap. 3, a solver that is based on quadratic approximation has been proposed. Recall that this method involves the construction of a grid of auxiliary set-points, for which the corresponding central costs are evaluated. Then, a quadratic approximation is performed in order to obtain an analytical form, from which the optimal set-point can be computed. However, the drawback of this method is that this approach becomes cumbersome when the dimension of the set-point vector r increases. Hence, the objective of this section is to propose a simpler solver that is based on gradient descend method.

This method is basically similar to the one that is described in Sect. 4.2. However, in order to ensure that this chapter is self-contained, we would like to reformulate this solver for this particular problem. Hence, let us recall the central optimization problem residing at the coordination layer.

$$r^{\text{opt}} = \underset{r\in\mathcal{R}}{\operatorname{argmin}}\, J_c(r) \tag{5.39}$$

where $\mathcal{R}$ is the admissible domain of set-points. In the gradient-based method, the problem

Algorithm 6 Anderson method with restarts. In the description of the algorithm, $g(\boldsymbol{v}^{in,(\sigma)}) = G(\boldsymbol{v}^{in,(\sigma)} - \boldsymbol{v}^{in,(\sigma)}$, $\Delta\boldsymbol{v}^{in,(i)} = \boldsymbol{v}^{in,(i+1)} - \boldsymbol{v}^{in,(i)}$, $g_i = g(\boldsymbol{v}^{in,(i)})$, $\Delta g_i = g_{i+1} - g_i$, $\mathcal{V}_i = \left[\Delta\boldsymbol{v}^{in,(i-m_\sigma)}, ..., \Delta\boldsymbol{v}^{in,(i-1)}\right]$ and $\mathcal{G}_i = \left[\Delta g_{i-m_\sigma}, ..., \Delta g_{\sigma-1}\right]$

```
Initialize:
    $\boldsymbol{v}_s^{in,(0)}; \leftarrow 0,\ s = 1, \ldots, n;$
    $m > 0;\ \sigma \leftarrow 0; c \leftarrow 0;\ \epsilon \leftarrow \infty;$
Coordinator sends $r_s$ to the subsystems;
while $(\sigma \leq \sigma_{\max})$ and $(\epsilon \leq \epsilon_{\max})$ do
    for $s \leftarrow 1, \ldots, n_s$ do                    ▷ Parallel operation performed by the subsystems
        Subsystem $s$ computes $\hat{\boldsymbol{v}}_s^{out}$ and sends to coordinator;
    end for
                                          ▷ The operations below are performed by the coordinator
    Coordinator forms up $\hat{\boldsymbol{v}}^{out,(\sigma)} := \operatorname*{col}_{s \in \mathcal{N}} \hat{\boldsymbol{v}}_s^{out,(\sigma)};$
    $m_\sigma = \min(m, c);$
    $\hat{\boldsymbol{v}}^{in,(\sigma)} = G_{in} \cdot \hat{\boldsymbol{v}}^{out,(\sigma)};$
    $g_\sigma = \hat{\boldsymbol{v}}^{in,(\sigma)} - \boldsymbol{v}^{in,(\sigma)};$
    if $\sigma == 0$ then
        $\boldsymbol{v}^{in,(\sigma+1)} = \hat{\boldsymbol{v}}^{in,(\sigma)};$
        $\Delta\boldsymbol{v}^{in,(\sigma)} = \boldsymbol{v}^{in,(\sigma+1)} - \boldsymbol{v}^{in,(\sigma)};$
    else
        $\Delta g_\sigma = g_\sigma - g_{\sigma-1};$
        $\mathcal{G}_\sigma = [\Delta g_{\sigma-m_\sigma}, \ldots, \Delta g_{\sigma-1}];$
        $\mathcal{V}_\sigma = \left[\Delta\boldsymbol{v}^{in,(\sigma-m_\sigma)}, ..., \Delta\boldsymbol{v}^{in,(\sigma-1)}\right];$
        Coordinator gets $\gamma^{(\sigma)}$ by solving (5.37);
        $\boldsymbol{v}^{in,(\sigma+1)} = \boldsymbol{v}^{in,(\sigma)} + g_\sigma - (\mathcal{V}_\sigma + \mathcal{G}_\sigma) \cdot \gamma^{(\sigma)};$
        $\Delta\boldsymbol{v}^{in,(\sigma)} = \boldsymbol{v}^{in,(\sigma+1)} - \boldsymbol{v}^{in,(\sigma)};$
    end if

    if $c == m$ then                                              ▷ check for restart
        $c \leftarrow 1;$
    else
        $c \leftarrow c + 1;$
    end if
    $\sigma \leftarrow \sigma + 1;$
    $\epsilon \leftarrow \max(|\boldsymbol{v}^{in,(\sigma+1)} - \boldsymbol{v}^{in,(\sigma)}|, 0);$
end while
```

(5.39) is iteratively optimized using the following updating rule:

$$p^{(i+1)} = r^{(i)} - \gamma^{(i)} \cdot \Delta J_c\left(r^{(i)}\right) \tag{5.40}$$

$$r^{(i+1)} = \mathbf{Pr}\left(p^{(i+1)} + c \cdot \left(p^{(i+1)} - p^{(i)}\right), \mathcal{R}_{trust}\right) \tag{5.41}$$

with $\mathcal{R}_{trust} = \{r|\underline{\rho}_r \leq r \leq \overline{\rho}_r\}$ being the trust-region of the updated set-point. The gradient of the central cost with respect to r can be computed by using the finite difference approximations, namely:

$$\nabla J_c(r^{(i)}) = \left[\frac{\delta J_c\left(r^{(i)}\right)}{\delta r_1^{(i)}}, \ldots, \frac{\delta J_c\left(r^{(i)}\right)}{\delta r_{n_r}^{(i)}}\right]^T \tag{5.42}$$

$$\frac{\delta J_c(r^{(i)})}{\delta r_j^{(i)}} \approx \frac{J_c\left(r^{(i)} + h^{(j)}\right) - J_c\left(r^{(i)}\right)}{h} \quad \forall j = 1, \ldots, n_r \tag{5.43}$$

with a slight abuse of notation, the subscript j indicates the j^{th} element of the vector $r^{(i)}$. The differentiation step is denoted by $h \ll 1$. The vectors $h^{(j)} \in \mathbb{R}^{n_r}$ contain the elements such that its j-th element is h, while the others are 0.

The variable γ in (5.40) is updated by using the Barzilai-Borwein formula given below:

$$\gamma^{(i+1)} = \frac{\|(r^{(i+1)} - r^{(i)}) \cdot (\nabla J_c(r^{(i+1)}) - \nabla J_c(r^{(i)}))\|}{\|\nabla J_c(r^{(i+1)}) - \nabla J_c(r^{(i)})\|^2} \tag{5.44}$$

Finally, the algorithm 7 summarizes the entire solver described in this sub-section that is used to solve the central optimization problem.

5.6 Simulation-based results

5.6.1 Parameter setting

Recall that the central cost is the sum of all the local costs contributed by the local subsystems, namely:

$$J_c(r) = \sum_{s \in \mathcal{N}} J_s(r) \tag{5.45}$$

where $J_s(r)$ are the local cost of the subsystem.

For S_s with $s \in \{1, 4, 7\}$ that have outputs to track the desired set-points r_s^d:

$$J_s(r|r_s^d) = \sum_{i=0}^{N-1} \left\|y_s(k+i) - r_s^d\right\|_{Q_c^{(s)}} + \|u_s(k+i)\|^2_{R_c^{(s)}} \tag{5.46}$$

Algorithm 7 Gradient-descent-based solver for solving the central optimization problem

```
Initialize:
    $r^{(1)}$ is warm-started by using the previous solution of the last resolution

for $i \leftarrow 1, \dots, n_{\max}^{\text{central}}$ do
    $J_c(r^{(i)}) \leftarrow$ Coordinator computes the corresponding central cost of $r^{(i)}$
    \\Compute the gradient at the current optimizing vector
    for $j \leftarrow 1, \dots, n_r$ do
        $J_c(r^i) \leftarrow$ Coordinator computes the corresponding central cost of $r^{\text{tmp}}$
        $\frac{\delta J_c(r^{(i)})}{\delta r_j^{(i)}} \leftarrow \frac{J_c(r^{(i)}+h^{(j)})-J_c(r^{(i)})}{h}$
    end for
    $\nabla J_c(r^{(i)}) \leftarrow \left[ \frac{\delta J_c(r^{(i)})}{\delta r_1^{(i)}}, \dots, \frac{\delta J_c(r^{(i)})}{\delta r_{n_r}^{(i)}} \right]^T$

    \\Update optimizing set-point vector
    $p^{(i+1)} = r^{(i)} - \gamma^{(i)} \cdot \Delta J_c(r^{(i)})$
    $r^{(i+1)} = \mathbf{Pr}(p^{(i+1)} + c \cdot (p^{(i+1)} - p^{(i)}), \mathcal{R}_{trust})$

    \\Compute the updating step for the next iterate
    $\gamma^{(i+1)} = \frac{\|(r^{(i+1)} - r^{(i)}) \cdot (\nabla J_c(r^{(i+1)}) - \nabla J_c(r^{(i)}))\|}{\|\nabla J_c(\boldsymbol{r}^{(i+1)}) - \nabla J_c(r^{(i)})\|^2}$
end for
\\Project the terminal optimizing vector on its admissible set
$r^{opt} = \mathbf{Pr}(r^{opt}, \mathcal{R})$
```

where $Q_c^{(s)} \in \mathbb{R}^{n_y^{(s)}}$ and $R_c^{(s)} \in \mathbb{R}^{n_u^{(s)}}$ are central-cost-related weighting matrices on outputs and inputs, which are listed below:

$$Q_c^{(1)} = \begin{bmatrix} 10^4 & 0 \\ 0 & 5 \cdot 10^4 \end{bmatrix} \qquad R_c^{(1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{5.47}$$

$$Q_c^{(4)} = 10^5 \qquad R_c^{(4)} = 0 \tag{5.48}$$

$$Q_c^{(7)} = 5 \cdot 10^3 \qquad R_c^{(7)} = 0 \tag{5.49}$$

For S_8 that has outputs not only to track its desired set-points r_8^d but also to satisfy the operational constraints which are 1 bar $\leq \mathrm{P}^L \leq$ 1.1 bar and 12 bar $\leq \mathrm{P}^H \leq$ 18 bar, its local cost is defined as follows:

$$\begin{aligned} J_8(r|r_8^d, \overline{y}_8, \underline{y}_8) = \sum_{i=0}^{N-1} & \left\|y_8(k+i) - r_8^d\right\|_{Q_c^{(8)}} + \|u_8(k+i)\|^2_{R_c^{(8)}} \\ & + \left\|\max(y_8(k+i) - \overline{y}_8, 0)\right\|_{Q_{cstr}^{(8)}} \\ & + \left\|\max(\underline{y}_8 - y_8(k+i), 0)\right\|_{Q_{cstr}^{(8)}} \end{aligned} \tag{5.50}$$

where the weighting matrices are given below:

$$Q_c^{(8)} = \begin{bmatrix} 10^6 & 0 \\ 0 & 5 \cdot 10^3 \end{bmatrix} \quad R_c^{(8)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad Q_{cstr}^{(8)} = \begin{bmatrix} 10^8 & 0 \\ 0 & 10^8 \end{bmatrix}$$

The other subsystems S_s with $s \in \{2, 3, 5, 6\}$ do not have any special operational criteria, their local costs are set to be null:

$$J_s(r) = 0 \tag{5.51}$$

For the local controller configuration. Table 5.3 summarizes the controller coefficients and the control updating constant τ_u of the involved subsystems.

In order to facilitate the result interpretation, some previously defined performance indicators in the last chapter will be needed. First, the closed-loop performance indicator J_c^{CL} is recalled, namely:

$$\begin{aligned} J_{sim} = \frac{1}{N_{sim}} \sum_{s \in \mathcal{N}} \sum_{i=1}^{N_{sim}} \Big[& \left\|y_s^{sim}(i) - r_s^d(i)\right\|_{Q_c^{(s)}} + \left\|u_s^{sim}(i)\right\|_{R_c^{(s)}} \\ & + \left\|\max(y_s^{sim}(i) - \overline{y}_s, 0)\right\|_{Q_{cstr}^{(s)}} \\ & + \left\|\max(\underline{y}_s - y_s^{sim}(i), 0)\right\|_{Q_{cstr}^{(s)}} \Big] \end{aligned} \tag{5.52}$$

Table 5.3: Coefficients of the local controllers of the subsystems.

Subsystem	Controller type	Coefficients	$\tau_u^{(s)}$
S_1	MPC	$Q = \text{diag}([100\ 100])$ $R = \text{diag}([1\ 1])$	5 s
S_4	PI	$K_p^4 = 3$ $K_i^4 = 0.1$	5 s
S_7	PI	$K_p^7 = 1$ $K_i^7 = 0.05$	5 s
S_8	Split range + PI	$K_p^{8,P^L} = 100$ $K_i^{8,P^L} = 10$ $K_p^{8,P^H} = 50$ $K_i^{8,P^H} = 1$	1 s

where the weighting matrices $Q_c^{(s)}$, $R_c^{(s)}$ are chosen to be identical to the weighting matrices defined for the central local costs. The maximum terminal error is also recalled, namely:

$$\overline{\epsilon}(k) := \max_{i=1}^{n_{ev}}(\epsilon_{end}^{(i)}) \tag{5.53}$$

where $\epsilon_{end}^{(i)}$ are the terminal convergence errors resulted from every central cost evaluation processes.

Finally, the terminal criteria on maximum iteration $\sigma_{\max}$ and $\epsilon_{\max}$ are respectively set at $\sigma_{\max} = 200$ and $\epsilon_{\max} = 1 \cdot 10^{-7}$.

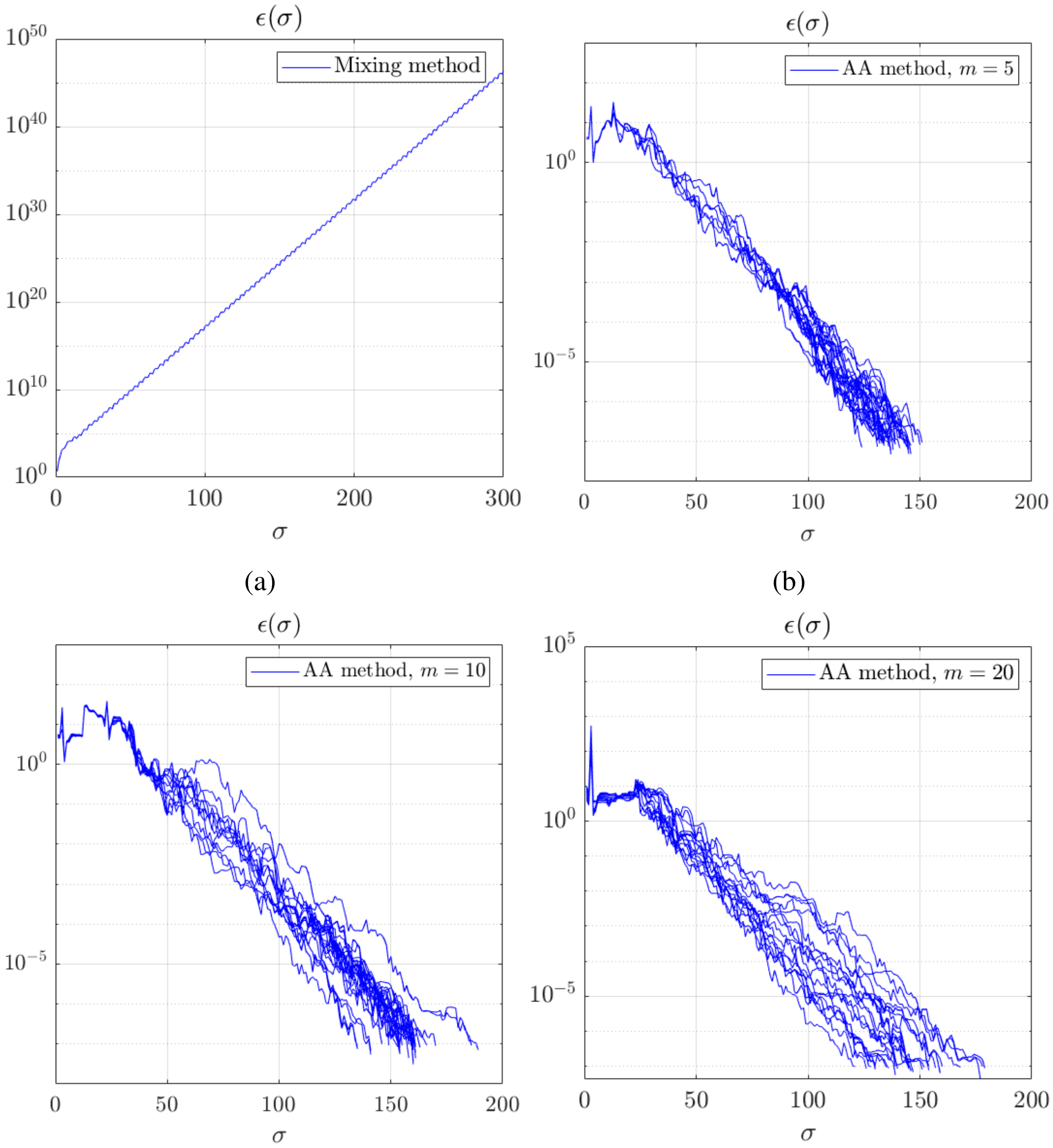
5.6.2 Numerical results

In this subsection, the results will be presented in order to evaluate:

- The efficiency of the Anderson acceleration method,
- The efficiency of the fixed-point based hierarchical control in the full cryogenic plant.

5.6.2.1 On the use of Anderson acceleration method in the fixed-point iteration

The convergence rate of the fixed-point iteration resulted from the communication between the coordinator and the subsystems is depicted in Fig. 5.5. This figure compares the convergence rates resulted by using the mixing method described in Chap. 3 and the Anderson acceleration method described in 5.5.1. The subfigure (1,1) shows that the mixing method can not ensure the convergence of the fixed-point iteration since the synthesized filtering matrix Π is not compatible with the actual controller information, which take into account the nonlinearities of the models and saturation constraints in their formulations. On the other hand, the convergence of the fixed-point iterations are enhanced, which is illustrated by subfigure (1,2), subfigure (2,1) and subfigure (2,2) for several memory length m. Crucially, the Anderson method makes the whole framework more modular since no apriori information of the subsystems is required for the coordinator to converge the fixed-point iteration.



(a) (b)

(c) (d)

Figure 5.5: convergence error of the fixed-point iterations resulted by using the mixing method described in Chap. 3 and the Anderson method for several choices of memory length m. It can be seen that the fixed-point iterations can not converge with the mixing method, while the AM method improves the convergence.

5.6.2.2 Closed-loop performance of the fixed-point-based hierarchical control method

The closed-loop performance of the framework will be presented in this section. Figure 5.6 shows the output behaviors of the system under the hierarchical control and the decentralized control as well as their closed-loop performance indices J_{sim} and the terminal convergence error $\bar{\epsilon}$. These strategies are simulated in a realistic scenario where the disturbance $\mathrm{NCR}_{22}^{(w)}$ (subfigure (3,1)) is applied to the plant. It can be noted that the liquid helium level Ltb_{131}(subfigure (1,1)), temperature Ttb_{130} (subfigure (1,2)) and the low pressure P^L (subfigure (2,2)) are better controlled than those given by the decentralized control, as their chosen weights on these outputs is higher than the others. Note that the high pressure P^H (subfigure (2,3)) is decreased by the coordinator to reduce the flowrate passing through the valves Cv_{157} and Cv_{156}, which will eventually prevent the increase of low pressure P^L. The closed-loop performance J_{sim} is illustrated in the subfigure (3,3). Furthermore, terminal convergence error $\bar{\epsilon}$ of the fixed-point iterations processed in the algorithm are enhanced by the AM method to be less than the defined limit $\epsilon_{\max}$.

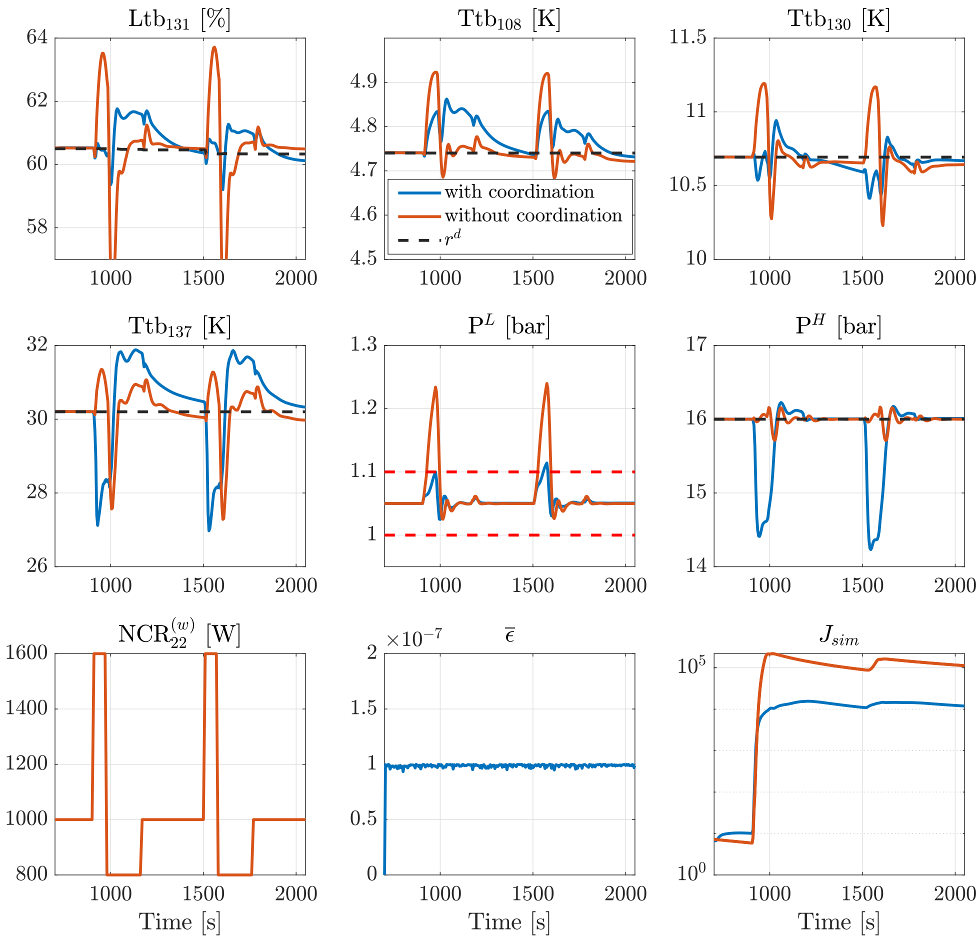


Figure 5.6: Comparison of the output behaviors of the system under the coordination and without the coordination.

For the actuators, Fig. 5.7 shows the manipulated input behaviors of the plant under the coordination and without the coordination in the same disturbance scenario. It can be seen that to satisfy the constraint on the low pressure P^L, the valves Cv^{156} and Cv^{157} are closed (corresponding to the increase of the pressure drop) under the coordination. The decharge valve Cv_{dch} is opened to decrease the high pressure P^H, which will also reduce the flow rate passing through the valves Cv^{156} and Cv^{157} at the inlets of the turbines.

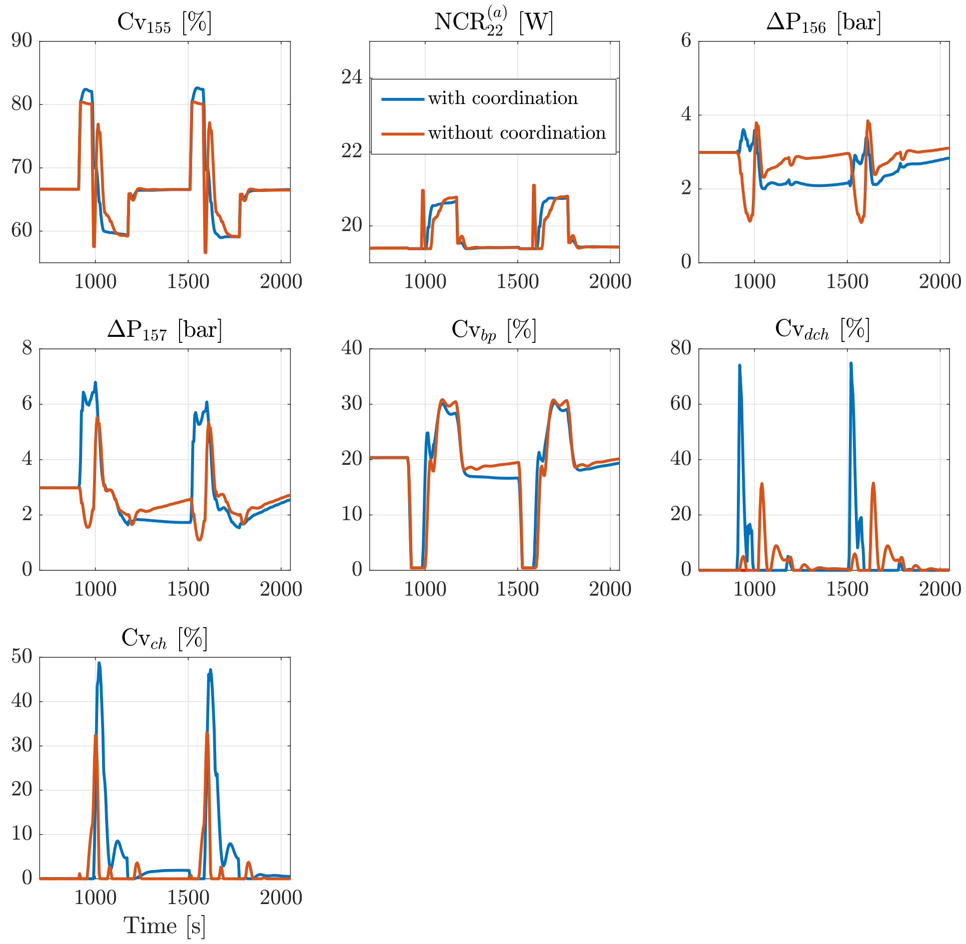


Figure 5.7: Manipulated input behaviors of the system under the coordination and without the coordination.

Finally, Fig. 5.8 shows the computation time of the subsystems that have controllers. Recall that the subsystem S_8 is assumed to have two processors that operate under two different updating periods and for two distinct tasks. The computation times shown in Fig.5.8 are the processor times of the subsystem involved in the resolution of the central problem, which do not exceed the update period $\tau_u^{\text{slow}} = 5\ s$. The processor time associated to the computation of the control input of $S_8^{(2)}$ at every updating control instant $k \cdot \tau_u^{\text{fast}}$ (with $k \in \mathbb{N}$ and $\tau_u^{\text{fast}} = 1\ s$) is negligible since the PI controller, whose computation involves only basic mathematical operations, is used.

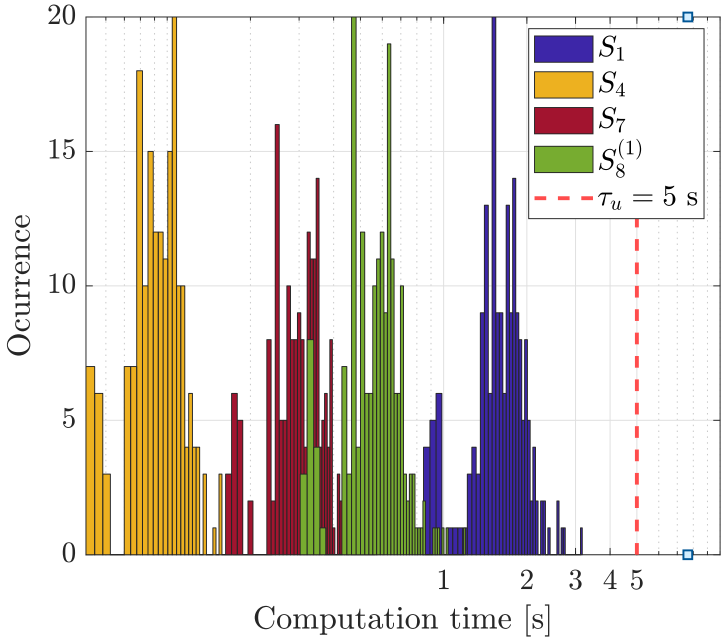


Figure 5.8: Computation time of the subsystems that have controllers.

5.7 Conclusion

In this chapter, the hierarchical control framework is applied to a more complex system where the number of subsystems is eight and the coupling topology is more complicated. Some developments of the algorithm have been made. To make the method modular, a residual-based iterative method, called Anderson acceleration, has been implemented to converge the fixed-point iteration by using only historical information during the iterations. In addition, a gradient-based optimization solver has been also implemented at the coordination layer to replace the quadratic approximation-based method described in Chap. 3 to find the optimal set point. Finally, the performance given by the method and by the developments is evaluated via numerical simulations, which shows that the hierarchical approach outperforms the decentralized one.

Conclusion and perspectives

Conclusion

The cryogenics process are often composed of several modules that span over a large area and are controlled individually with basic knowledge of the coupling effects between them. For instance, the grand refrigerator at CEA, which has a cooling power of 400 W at 1.8 K, has a warm compression station and a cold box. Each of them is controlled by at least one local controller, which could be seen as a classic decentralized control method, in which these local controllers operate individually. This method is often easy to be implemented but do not achieve high stability and optimal economic performance.

Regarding of this fact, this book has proposed and promoted by simulation results a hierarchical control framework, which is based on fixed-point iterations, in order to control the 400W @ 1.8 K refrigerator at CEA. In this framework, there exist two layers which are the coordination layer and the local layer. In the lower layer, the local controllers are implemented in order to control the controlled subsystems. In the coordination layer, the coordinator is designed in order to pilot the local controllers by sending the set-points that optimize some defined operation cost. To do so, an algorithm that is based on fixed-point iteration is proposed. More precisely, for a given set-point sent from the coordinator to the subsystems, the local controllers communicate extensively with the coordinator by sending the estimated coupling profiles over a prediction horizon, which will then be updated by the coordinator and be sent back to the subsystems. The associated operation cost to the given set-point can be computed if the communication iterations between the coordinator and the subsystems converge to a set of coupling profiles. Having an algorithm that is capable of estimating the global cost for any given set-points, while taking into account the coupling effects, a simple optimization algorithm that is based on quadratic approximation is used to iteratively find the optimal set-point.

Chapter 3 recalled the mentioned hierarchical control framework and applied it to the control of the large refrigerator, where nonlinearities and real-time implementation are considered. Specifically, local controllers, which take into account actuator saturation constraints and subsystem model nonlinearities, are incorporated into the framework. The control performance was compared using simulations, which showed that nonlinear controllers can be integrated into this framework and give better performance than using linear controllers. However, the use of nonlinear controllers can increase the computational load and make the entire framework inapplicable in real-time applications. Therefore, an optimization distribution method is proposed to reduce the computational burden, which must be performed in a single control updating period. Specifically, by cyclically updating the optimization variable (setpoint variable) over time, the number of costs to be evaluated can be reduced, and thus the computation time. The simulation results showed that the performance obtained with the proposed optimization distribution method is better than that given by the decentralized control method and is not seriously degraded when compared to the full optimization approach.

Chapter 4 proposed some avenues to follow when using NMPCs poses a computational time problem. First, it is proposed to use a simple optimization solver based on the gradient descent method instead of a high-performance solver, but with a higher computational cost. Second, machine learning approaches should be considered when the controlled process operates only in certain operating regimes. Specifically, deep neural networks are used to approximate the NMPC laws, thus reducing the computational burden. As a result, control actions can be updated more frequently and control performance can be improved.

Chapter 5 described some advances that are needed in order to apply the proposed framework to more complex cryogenic systems and to satisfy the modular privacy preservation requirement, which was defined as a design criterion from the beginning. The first advance aims to replace the model-based filter with an algorithm that relies solely on historical residual data to converge the fixed-point iterations. The second advance is to use a simple optimization solver based on the gradient descent method to solve the central problem delivering the optimal set-point. The effectiveness of the proposed hierarchical control framework, combined with these advances, is assessed by simulation results and found to be better than the full decentralized control, while being real-time implementable.

Perspectives

In the future, the proposed framework will be applied on an even more complicated cryogenic refrigerator at CERN. In this study, the computation of the coordination layer and the local layer is handled on a single computer, which is not the intention of this framework. Indeed, local control in real life should be implemented in a distributive way and correspond to their subsystems, which means there should be an individual processor computing the control profiles and the coupling profiles for each subsystem. The effects induced by the communication between the local agents and the coordinator such as delays, information loss, etc. should be taken into account for a real implementation.

The proposed framework could also be modified. Indeed, at the local layer, each subsystem which have decision variable u_s is controlled by a controller. Then, the coordinator tries to solve the central optimization problem recalled as follows:

$$r^{opt} = \underset{r}{\operatorname{argmin}}\, J_c(r, \boldsymbol{v}^{in}) \tag{5.54}$$

$$\text{subject to: } \boldsymbol{v}^{in} = G_{in} \cdot \boldsymbol{g}_{out}(r, v^{in}) \tag{5.55}$$

where the optimizing variable is the set-point vector r.

Keeping the same spirit of the previous works, another hierarchical control framework can be developed while keeping the assumption that the coordinator ignores all the mathematical models of the subsystems. Let us re-define the process for this framework.

Assumption 1

Each subsystem S_s receive from the coordinator:

- *a presumed incoming profile $\boldsymbol{v}_s^{in}$ and*
- *a given* ***control profile*** *$\boldsymbol{u}_s$ (required if $s \in \mathcal{N}^{ctr}$),*

so that S_s can process an algorithm to compute what would be:

- *Its resulting outgoing profile $\boldsymbol{v}_s^{out}$ and*
- *Its contribution J_s to the central cost*

The central cost is assumed to be of the form:

$$J_c(\boldsymbol{u}, \boldsymbol{v}^{in}) := \sum_{s\in\mathcal{N}^{ctr}} J_s(\boldsymbol{u}_s, \boldsymbol{v}_s^{in}) + \sum_{s\in\mathcal{N}^{unc}} J_s(\boldsymbol{v}_s^{in})$$

where $\boldsymbol{u} := \underset{s\in\mathcal{N}^{ctr}}{col}\ \boldsymbol{u}_s$ and $\boldsymbol{v}^{in} := \underset{s\in\mathcal{N}}{col} \boldsymbol{v}_s^{in}$

Consequently, the optimization problem that the coordinator needs to solve is redefined below:

$$\boldsymbol{u}^{opt} = \underset{\boldsymbol{u}}{\text{argmin}}\, J_c(\boldsymbol{u}, \boldsymbol{v}^{in})$$
$$\text{subject to: } \boldsymbol{v}^{in} = G_{in} \cdot \boldsymbol{g}_{out}(\boldsymbol{u}, v^{in})$$

Indeed, the above problem can be considered as a modular oriented NMPC problem. However, this approach simplifies the modeling step for large-scale systems, especially for the large-scale cryogenic refrigerator where the subsystems are strongly coupled with their neighbors.

The proposed framework should not be limited to cryogenic applications. It could be developed to be easily applied to general cases such as electrical systems or water supply systems. Further developments could start to investigate and implement a method capable of decomposing the overall system into many subsystems with coupling signals. Indeed, in our study, we do not need such a method to decompose the refrigerator since the physical principles of coupling effects between subsystems are known. However, to make the proposed framework more generic, a system decomposition method is needed. Moreover, the framework can be developed to be dedicated to cases where controllers such as PID, LQR, MPC and the coordinator can be generated automatically, which makes this method user-friendly.

APPENDIX A

Modeling Turbine and Joule-Thomson cycle

A.1 Turbine

In this section, the mathematical model of a turbine will be described. Fig. A.1 shows the input and outputs of a cryogenic turbine. The model express the output temperature T^H_{out} , the input flow rate M^H_{out} and the output flow rate M^L_{out} of the turbine in the function of the thermodynamic of the input fluid defined by the input temperature T^H_{in}, the high pressure P^H_{in} and the low pressure P^L_{in}.

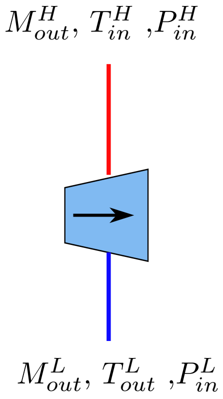


Figure A.1: Synoptic view of a turbine with its input and output variables denoted respectively by the grandeurs with exponent *in* and *out*.

It is assumed that the turbine does not stock gas and mass and it does not have any thermal capacity or inertia. According to [66], the flow rate through a cryogenic turbine can be expressed as below:

$$M_{tb} = M^H_{out} = M^L_{out} = \frac{C \cdot P^H_{in} \cdot A}{\sqrt{\dfrac{Z \cdot R \cdot T^H_{in}}{M_{he}}}}, \quad C = \sqrt{\gamma \cdot \left(\frac{2}{\gamma + 1}\right)^{\dfrac{\gamma + 1}{\gamma - 1}}} \tag{A.1}$$

in which C denotes a constant depending on atomic proprieties of the fluid, A represents the section of the collar, Z represents the compressibility factor of the helium, R represents the perfect gas constant, M_{he} is the molar mass of helium and finally γ is the ratio of the specific heat.

The turbine produces cold with energy extraction. This cooling power is proportional to the enthalpy drop that occurs between the inlet and the outlet the outlet:

$$Q = M_{tb} \cdot (h^H - h^L) \tag{A.2}$$

where h^H and h^L denote the enthalpies of the fluids at the inlet and outlet of the turbine. The enthalpy of the input fluid can be computed by using the bibliography *Hepak*'s function: $h^H = hecalc('H', P_{in}^H, T_{in}^H)$. On the other hand, if there is no friction in the turbine, the enthalpy of the fluid at the outlet could be computed similarly by using the pressure P_{in}^L and the input entropy s^H, but it is not the case.

Let us consider an isentropic efficiency η. The extracted power is then written :

$$Q = M_{tb} \cdot \eta \cdot (h^H - h^L) \tag{A.3}$$

where h^L represents the enthalpy of the fluid at the turbine's outlet if the expansion is isentropic. It could be computed by using the functions of Hepak as shown below:

$$s^H = hecalc('S', P_{in}^H, T_{in}^H) \tag{A.4}$$

$$h_{is}^L = hecalc('H', P_{in}^L, s^L = s^H) \tag{A.5}$$

Then, the enthalpy at the outlet is computed as follows:

$$h^L = h^H(1 - \eta) - \eta h_{is}^L \tag{A.6}$$

Finally, the temperature at the outlet of the turbine is compute by using a Hepak's function:

$$T_{out}^L = hecalc('T', h^L, P_{in}^L) \tag{A.7}$$

For the implementation, all the variables that are computed by the Hepak's functions could be approximated by some polynomial functions in interested ranges of involved parameters. Finally, the output of the turbine can be expressed as an equation below:

$$y_{tb} = g_{tb}(v_{tb}^{in}) \text{ with } y_{tb} = \begin{bmatrix} T_{out}^L \\ M_{out}^L \\ M_{out}^H \end{bmatrix} \text{ and } v_{tb}^{in} = \begin{bmatrix} T_{in}^H \\ P_{in}^H \\ P_{in}^L \end{bmatrix} \tag{A.8}$$

where y_{tb} and v_{tb}^{in} represent represent the output vector and limit condition of the turbine.

A.2 Joule-Thomson cycle

A.2.1 J-T Valve

The controlled valves are used in order to control pressures, flow rates or even to control the process. The valves that are modeled are the proportional valves that are used in order to control the flow rate.

Assuming that the fluid is not stocked in the valve and the time to position the needle is negligible. The process occurs in the valve is assumed to be completely isenthalpic. These assumptions allow to model the valve as an static function.

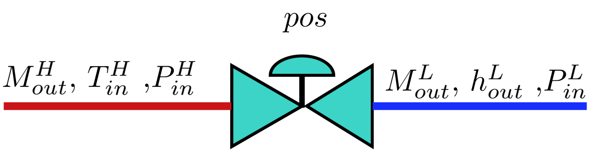


Figure A.2: Synoptic view of a valve with its input and output variables denoted by the grandeurs with subscripts *in* and *out* respectively.

Fig. A.2 shows the input and output of the modeling function. The model expresses the inlet flow rate, the outlet flow rate and the outlet temperature as a function of the inlet temperature, the inlet pressure and the outlet pressure.

The inlet and outlet flow rates are assumed to be equal because the valve does not store the fluid.

$$M = M^H_{out} = M^L_{out} \tag{A.9}$$

where M denotes the flow rate passing through the controlled valve. According to [67], this flow rate can be computed as follows:

$$M = 7.59 \cdot 10^{-3} \cdot CV \cdot \left(1 - \frac{X}{3 \cdot X_C}\right) \sqrt{\rho^H_{in} \cdot P^H_{in} \cdot X} \tag{A.10}$$

$$CV = \frac{CV_{\max}}{R_v} \left(\exp\left(\frac{pos}{100} \log R_V\right) - \left(1 - \frac{pos}{100}\right)\right) \tag{A.11}$$

$$X = \min\left(\frac{P^H_{in} - P^C_{in}}{P^H_{in}}, X_C\right), \quad X_C = \frac{\gamma}{1.4} X_t \tag{A.12}$$

where γ denotes the specific heat ratio and ρ^H_{in} denotes the volumic mass of the inlet fluid. $CV_{\max}$ and R_V are the dimensional constants of the valve, which are respectively the flow rate coefficient and the rangeability. X_t is a design constant given by the supplier and pos is the opening position of the valve.

Indeed, the process occurs in the valve is isenthalpic, the enthalpy of the fluid at the inlet and outlet of the valve are equal and computed by the Hepak's function, namely:

$$h^L = h^H = hecalc('H', T_{in}^H, P_{in}^H) \tag{A.13}$$

$$\tag{A.14}$$

It is essential to note that the variables, which are computed by using the Hepak's functions, can be approximated by some polynomial functions in the interested range of corresponding parameters, namely:

$$h^L = h^H = f_v^H(T_{in}^H, P_{in}^H) \tag{A.15}$$

$$\tag{A.16}$$

Finally, all the output of the modeling function are described. The equation form of the valve is given below:

$$y_{vlv} = g_{vlv}(u_{vlv}, v_{vlv}^{in}) \text{ with } y_{vlv} = \begin{bmatrix} M_{out}^H \\ M_{out}^L \\ h^L \end{bmatrix} \text{ and } v_{vlv}^{in} = \begin{bmatrix} T_{in}^H \\ P_{in}^H \\ P_{in}^L \end{bmatrix} \quad u_{vlv} = pos \tag{A.17}$$

A.2.2 Helium bath

The helium bath is where the helium fluid is partially liquefied and rests in the bath, while the gaseous part leaves and returns to the cycle. Fig. A.3 shows the input and output of the function modeling the helium bath. The model expresses the helium level, the outlet temperature, the inlet and outlet pressure as a function of the inlet pressure, the inlet flow rate and the inlet temperature.

Let's assume that the total mass of the fluid (gas + liquid) is punctual. This type of assumption implicates that the physique phenomena related to the non-uniformity of the fluid properties in the bath such as the stratification or the natural convection are not modelized.

The thermodynamic state of the fluid contained in the bath can be derived by using the conservation laws of energy and mass, namely:

$$\dot{\rho} = \frac{\sum M_{in}^H - \sum M_{in}^L}{Vol} \tag{A.18}$$

$$\dot{u} = \frac{\sum \varphi_{in}^H - \sum \varphi_{in}^L + \sum P}{\rho \cdot Vol} - u \cdot \frac{\dot{\rho}}{\rho} \tag{A.19}$$

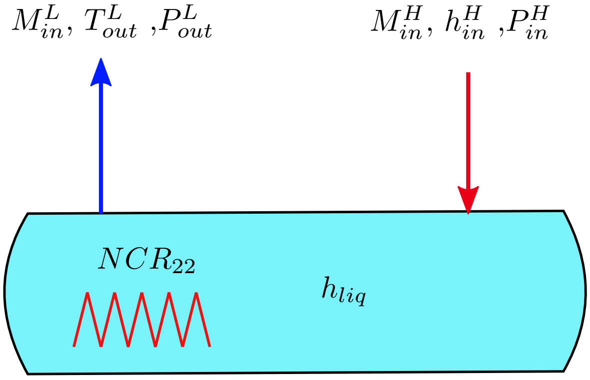


Figure A.3: Synoptic view of a helium bath with its input and output variables denoted by the grandeurs with subscripts *in* and *out* respectively.

where ρ denotes the volumic mass, Vol denotes the total volume of the bath, $\sum M_{in}^H$ and $\sum M_{in}^L$ represent respectively the sums of the incoming and the outgoing flow rates. $\sum \varphi_{in}^H$ and $\sum \varphi_{in}^L$ represent respectively the sums of the incoming and the outgoing enthalpy fluxes of the bath ($\sum \varphi_{in}^H = M_{in}^H \cdot h_{in}^H$ and $\sum \varphi_{in}^L = M_{in}^L \cdot h_{in}^L$). Finally, $\sum P$ denotes the sum of the thermal charges received by the bath. In our configuration, this variable is the sum of the power of the heating actuator $NCR_{22}^{(a)}$ and the heating disturbance $NCR_{22}^{(w)}$.

The pressure at the outlet of the bath is computed by the Hepak's function, namely:

$$P_{out}^L = hecalc('P', u, \rho) \tag{A.20}$$

Let's assume that there is one way for the gaseous helium to exit the bath and return to the cycle.

While the incoming enthalpy flux is imposed on the bath, the outgoing enthalpy flux imposed by the bath is computed by using the Hepak's function:

$$h_{out} = hecalc('H', P_{out}^L) \tag{A.21}$$

The helium level in the bath can be computed by the equation below:

$$h_{liq} = \left((1-q) \cdot \frac{\rho}{\rho_{liq}} \cdot \frac{z_{\max}}{pos_{\max} - pos_{\min}} - \frac{pos_{\min}}{pos_{\max} - pos_{\min}} \right) \cdot 100\% \tag{A.22}$$

where $z_{\max}$ is the maximum height of the bath, $pos_{\max}$ and $pos_{\min}$ are respectively the position of the upper and lower ends of the level detector. The density of the liquid helium can be deduced via the pressure in the bath by using the Hepak's function, namely:

$$\rho_{liq} = hecalc('\rho', P_{out}^L, 'SL') \tag{A.23}$$

The helium quality q appearing in (A.22) is derived in the equation below:

$$q = \frac{\rho_{gas}}{\rho} \cdot \frac{\rho_{liq} - \rho}{\rho_{liq} - \rho_{gaz}} \tag{A.24}$$

with the density of the gaseous helium being computed by:

$$\rho_{gas} = hecalc('\rho', P_{out}^{L},' SV') \tag{A.25}$$

Finally, the dynamic of this component can be expressed by the equations below:

$$\dot{x}_{sp} = f_{sp}(x_{sp}, u_{sp}, w_{sp}, v_{sp}^{in}) \tag{A.26}$$

$$y_{sp} = g_{sp}(x_{sp}) \tag{A.27}$$

with

$$x_{sp} = \begin{bmatrix} \rho \\ u \end{bmatrix}, \quad w_{sp} = NCR_{22}^{(w)}, \quad u_{sp} = NCR_{22}^{(a)}, \tag{A.28}$$

$$v_{sp}^{in} = \begin{bmatrix} M_{in}^{H} \\ M_{in}^{L} \\ H_{in}^{H} \end{bmatrix}, \quad y_{sp} = \begin{bmatrix} P_{out}^{L} \\ P_{in}^{H} \\ T_{out}^{L} \\ h_{liq} \end{bmatrix} \tag{A.29}$$

A.2.3 Heat exchanger

In order to modelize the heat exchanger, some assumptions need to be made:

1. channels with the same origin/termination will be considered as one:
2. only one spatial coordinate will be considered, which is the fluid direction;
3. the pressures will be considered linearly decreasing in the direction of the fluid;
4. the walls of the exchanger are considered adiabatic;
5. the longitude thermal conductivity will be considered null;
6. only one state variable will be considered per branch.

These assumptions have been validated in [68] by experimental results.

The interested heat exchanger in this study is the exchanger with two fluids in counterflow. The model expresses the temperatures, flow rates and pressures at the outlet in function of the temperatures, flow rates and pressures at the inlet of the heat exchanger. The behavior of the heat exchanger can be approximated by the spatial discretization as proposed in [69, 70, 71]. Indeed, the heat exchanger can be decomposed in N elementary components. Each zone is affected by the parameters corresponding to the current state of the fluid.

$$\left(\frac{\rho^H Cp_i^H V^H + \mathcal{M}^{al} Cp^{al}/2}{N}\right)\dot{T}_i^H = M^H Cp_i^H (T_{i-1}^H - T_i^H) - Q_i \tag{A.30}$$

$$\left(\frac{\rho^L Cp_i^L V^L + \mathcal{M}^{al} Cp^{al}/2}{N}\right)\dot{T}_i^L = M^L Cp_i^L (T_{i-1}^L - T_i^L) + Q_i \tag{A.31}$$

$$Q_i = h\left(\frac{T_{i-1}^H + T_i^H}{2} - \frac{T_{i-1}^L + T_i^L}{2}\right) \tag{A.32}$$

where $\rho^H/\ \rho^L$ and $Cp^H/Cp^L\ V^H$ denote the densities and specific heat capacities in heat exchanger's pipes under high pressure and low pressure, whose volumes are denoted by V^H/V^L, respectively. $\mathcal{M}^{al}$ and Cp^{al} represent the mass and the specific heat capacity of aluminum used to make the heat exchanger. T_i^H and T_i^L denotes respectively the temperatures under high pressure and low pressure at the limit of each zone. The distribution of these temperatures is illustrated in Fig.A.4. Q_i is the heat exchanged between the high pressure pipeline and low pressure pipeline, which is the difference between the mean temperatures of each zone multiplied by the heat exchange coefficient h.

The flow rates in each zone are considered to be given by:

$$M^H = K^H \cdot (P_0^H - P_N^H) \tag{A.33}$$

$$M^L = K^L \cdot (P_0^L - P_N^L) \tag{A.34}$$

where K^H and K^L are respectively the pressure loss coefficients of the high pressure and low pressure pipelines. if the heat exchanger is connected to the helium bath, the low pressure P_0^L is computed by:

The dynamic of the pressure at the outlet of the high pressure branch is given by:

$$\dot{P}_N^H = Kp^H \cdot (M^H - M_{in}^H) \tag{A.35}$$

where Kp is the loss coefficient and M_{in}^H is the flow rate imposed at the inlet of the high pressure branch.

Finally, the dynamic of the heat exchanger can be expressed by the following equations:

$$\dot{x}_{ex} = f_{ex}(x_{ex}, v_{ex}^{in}) \tag{A.36}$$

$$y_{ex} = g_{ex}(x_{ex}, v_{ex}^{in}) \tag{A.37}$$

where

$$x_{ex} = \begin{bmatrix} T_1^H \\ T_N^L \\ \vdots \\ T_N^H \\ T_1^L \\ P_N^H \end{bmatrix},\ v_{ex}^{in} = \begin{bmatrix} T_0^H \\ T_0^L \\ M_{in}^H \\ P_0^H \\ P_0^L \\ P_N^L \end{bmatrix},\ y_{ex} = \begin{bmatrix} T_N^H \\ T_N^L \\ M^H \\ M^L \\ P_N^H \end{bmatrix} \tag{A.38}$$

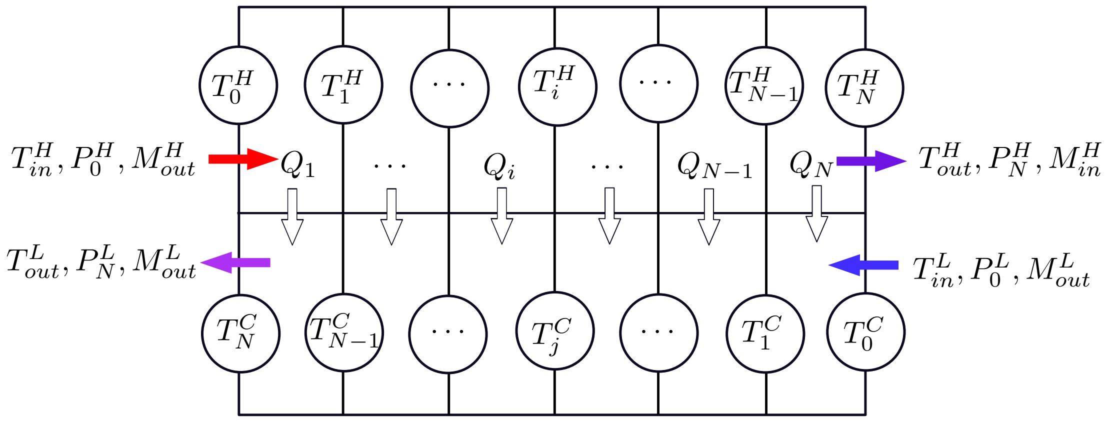


Figure A.4: Synoptic view of a heat exchanger with its input and output variables denoted by the grandeurs with subscripts *in* and *out* respectively. Equivently, $T_0^H = T_{in}^H$, $T_0^L = T_{in}^L$, $T_{out}^H = T_N^H$, $T_{out}^L = T_N^L$.

Note that the heat exchanger used in Chap. 4 is modelled with $N = 1$/

A.2.4 Model of Joule-Thomson cycle

At this stage, the three components of the J-T cycle have been modeled, which are now assembled in order to constitute the complete system. The components connect with each other following the rule depicted in Fig. A.5.

Then, the final model of the J-T cycle can be expressed by the following equations:

$$\dot{x}_{jt} = f_{jt}(x_{jt}, u_{jt}, w_{jt}, v_{jt}^{in}) \tag{A.39}$$

$$y_{jt} = g_{jt}(x_{jt}, u_{jt}, w_{jt}, v_{jt}^{in}) \tag{A.40}$$

where x_{jt} gathers the state vectors, u_{jt} gathers the actuator, w_{jt} gathers the disturbance and v_{jt}^{in} denotes the limit condition given by the previous subsystem, namely:

$$x_{jt} = \begin{bmatrix} x_{ex} \\ x_{ps} \end{bmatrix} \quad u_{jt} = u_{vlv} \quad w_{jt} = w_{ps} \tag{A.41}$$

Meanwhile, y_{jt} gathers the measured outputs which are the liquid helium level in the phase separator and the temperature at the oulet of the J-T valve, namely:

$$y_{jt} = \begin{bmatrix} h_{liq} \\ T_{vlv}^{out} \end{bmatrix} \tag{A.42}$$

As long as the model predictive controller is concerned, the system need be to discretized. The simplest method is the Runge-Kutta 4th method. However, in our case, the differential

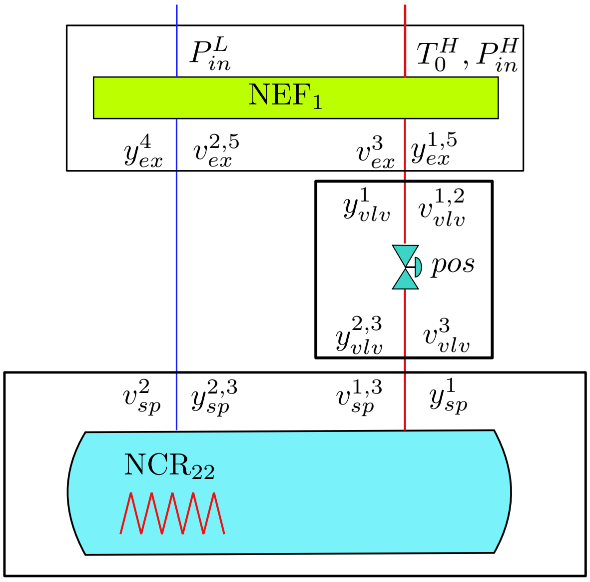


Figure A.5: Synoptic view of the J-T cycle with the variables connections. The coupling signal input v^{in} of a component is placed opposite with the output y of the component by which it is coupled. The upper script indicates the elements of the vector. In this figure, the variables $T_i^H n$ and T_{out}^H / $T_i^L n$ and T_{out}^L are respectively T_0^H and T_N^H / T_0^L and T_N^L in the analytic model (A.30)-(A.32)

equations (A.39) are stiff, which needs an discretizing step relatively small in comparing to the sampling time $T_s = 5s$ and the prediction horizon $N \cdot T_s$ (with $N = 100$). Effectively, this small discretizing step can overcome the divergence resulted from using the Runge-Kutta method, but induces some computational burdens.

Instead of using the Runge-Kutta method, an implicit solver is proposed to be used, whose formulation is given below:

$$x_{jt}^{+} = x_{jt} + \frac{\mathbb{I} - T_s \cdot \nabla_{x_{jt}} f_{jt}(x_{jt}, u_{jt}, w_{jt}, v_{jt}^{in})}{T_s \cdot f_{jt}(x_{jt}, u_{jt}, w_{jt}, v_{jt}^{in})} \tag{A.43}$$

$$= f_{jt}^{dsct}(x_{jt}, u_{jt}, w_{jt}, v_{jt}^{in}) \tag{A.44}$$

where $\mathbb{I}$ is the identity matrix with appropriate dimension and $\nabla_{x_{jt}} f_{jt}(x_{jt})$ is the jacobian matrix of $f_{jt}(x_{jt})$ with respect to x_{jt}.

Bibliography

[1] Mazen Alamir et al. “Fixed-point based hierarchical MPC control design for a cryogenic refrigerator”. In: *Journal of Process Control* 58 (2017), pp. 117–130.

[2] *Electricity production by source, World*. https://ourworldindata.org/grapher/electricity-prod-source-stacked. 2022.

[3] Landon Stevens et al. “The footprint of energy: Land use of US Electricity production”. In: *STRATA: Logan, UT, USA* (2017).

[4] Ottmar Edenhofer. *Climate change 2014: mitigation of climate change*. Vol. 3. Cambridge University Press, 2015, pp. 1306–1308.

[5] Arthur Stanley Eddington. *The internal constitution of the stars*. Cambridge University Press, 1988.

[6] Laurent Costanzo. “Etude expérimentale des aspects topologiques du divertor ergodique de Tore Supra”. PhD thesis. Aix-Marseille 1, 2001, pp. 32–33.

[7] T Kammash. “Fusion power and its prospects”. In: *Alternative Energy Sources*. Elsevier, 1981, pp. 607–685.

[8] Alexandre Bovet. “Suprathermal ion transport in turbulent magnetized plasmas”. PhD thesis. Feb. 2015. DOI: `10.5075/epfl-thesis-6527`.

[9] Gianmaria De Tommasi. “Plasma Magnetic Control in Tokamak Devices”. In: *Journal of Fusion Energy* 38 (Aug. 2019). DOI: `10.1007/s10894-018-0162-5`.

[10] P Roussel et al. “DESIGN STATUS OF THE CRYOGENIC SYSTEM AND OPERATION MODES ANALYSYS OF THE JT-60SA TOKAMAK”. In: *AIP Conference Proceedings*. Vol. 1218. American Institute of Physics. 2010, pp. 1445–1447.

[11] Benjamin Bradu, Ph Gayet, and S-I Niculescu. “Control optimization of a lhc 18 kw cryoplant warm compression station using dynamic simulations”. In: *AIP conference proceedings*. American Institute of Physics. 2010, pp. 1619–1626.

[12] Fanny Clavel et al. “An innovative control of a warm compression system for a Helium refrigerator subject to high pulsed loads”. In: *International Cryogenic Engineering Conference*. 2010.

[13] M Zhuang et al. “Control optimization of the cryoplant warm compressor station for east”. In: *AIP Conference Proceedings*. Vol. 1573. American Institute of Physics. 2014, pp. 1728–1734.

[14] Venkatarao Ganni and Peter Knudsen. “Optimal design and operation of helium refrigeration systems using the Ganni cycle”. In: *AIP Conference Proceedings*. Vol. 1218. American Institute of Physics. 2010, pp. 1057–1071.

[15] S Claudet et al. “Four 12kW/4.5 K cryoplants at CERN”. In: *Cryogenics* 34 (1994), pp. 99–102.

[16] Cecil L. Smith. “Split-Range Control”. In: *Advanced Process Control*. John Wiley & Sons, Ltd, 2010. Chap. 3, pp. 86–125. ISBN: 9780470612903.

[17] Fanny Clavel et al. “Multivariable control architecture for a cryogenic test facility under high pulsed loads: Model derivation, control design and experimental validation”. In: *Journal of Process Control* 21.7 (2011), pp. 1030–1039.

[18] F. Bonne, M. Alamir, and P. Bonnay. “Experimental investigation of control updating period monitoring in industrial PLC-based fast MPC: Application to the constrained control of a cryogenic refrigerator”. In: *Control Theory and Technology* 15 (2014), pp. 92–108.

[19] B Bradu et al. “CRYOLIB. A commercial library for modelling and simulation of cryogenic processes with EcosimPro”. In: (2012).

[20] Sunil Manohar Dash. “Study of Cryogenic Cycles with Aspen-Hysys Simulations”. PhD thesis. 2009.

[21] Xiao-Yen Wang et al. “Numerical Modeling of Thermal Stratification in Cryogenic Propellant Tanks”. In: *Thermal and Fluids Analysis Workshop (TFAWS)*. GRC-E-DAA-TN60294. 2018.

[22] F. Bonne et al. “Simcryogenics: a Library to Simulate and Optimize Cryoplant and Cryodistribution Dynamics”. In: *IOP Conference Series: Materials Science and Engineering* 755 (June 2020), p. 012076.

[23] Shih-Ho Wang and Edward Davison. “On the stabilization of decentralized control systems”. In: *IEEE Transactions on Automatic Control* 18.5 (1973), pp. 473–478.

[24] Edward J Davison and Timothy N Chang. “Decentralized stabilization and pole assignment for general improper systems”. In: *1987 American Control Conference*. IEEE. 1987, pp. 1669–1675.

[25] Jan Lunze. *Feedback control of large-scale systems*. Prentice Hall New York, 1992.

[26] Dragoslav D Siljak. *Decentralized control of complex systems*. Courier Corporation, 2011.

[27] M Ikeda, DD Šiljak, and DE White. “Decentralized control with overlapping information sets”. In: *Journal of optimization theory and Applications* 34.2 (1981), pp. 279–310.

[28] Masao Ikeda, DD Siljak, and DE White. “An inclusion principle for dynamic systems”. In: *1982 American Control Conference*. IEEE. 1982, pp. 884–892.

[29] Altuğ İftar. “Decentralized estimation and control with overlapping input, state, and output decomposition”. In: *Automatica* 29.2 (1993), pp. 511–516.

[30] Dragoslav D Šiljak and AI Zečević. “Control of large-scale systems: Beyond decentralized feedback”. In: *Annual Reviews in Control* 29.2 (2005), pp. 169–179.

[31] DD Šiljak. “Decentralized control and computations: status and prospects”. In: *Annual Reviews in Control* 20 (1996), pp. 131–141.

[32] Lubomir Bakule. “Decentralized control: An overview”. In: *Annual reviews in control* 32.1 (2008), pp. 87–98.

[33] Riccardo Scattolini. “Architectures for distributed and hierarchical Model Predictive Control - A review”. In: *Journal of Process Control* 19.5 (2009), pp. 723–731. ISSN: 0959-1524.

[34] R. R. Negenborn and J. M. Maestre. “On 35 Approaches for Distributed MPC Made Easy”. In: *Distributed Model Predictive Control Made Easy*. Ed. by José M. Maestre and Rudy R. Negenborn. Dordrecht: Springer Netherlands, 2014, pp. 1–37.

[35] Yan Zhang and Shaoyuan Li. “Networked model predictive control based on neighbourhood optimization for serially connected large-scale processes”. In: *Journal of process control* 17.1 (2007), pp. 37–50.

[36] Damien Trentesaux. “Distributed control of production systems”. In: *Engineering Applications of Artificial Intelligence* 22.7 (2009), pp. 971–978.

[37] Mehrdad Yazdanian and Ali Mehrizi-Sani. “Distributed control techniques in microgrids”. In: *IEEE Transactions on Smart Grid* 5.6 (2014), pp. 2901–2909.

[38] Natalia I Marcos, J Fraser Forbes, and Martin Guay. “Price-driven coordination of distributed MPC controllers for constrained dynamic systems”. In: *Industrial & Engineering chemistry research* 52.49 (2013), pp. 17451–17464.

[39] R Martı, D Sarabia, and C de Prada. “Price-driven coordination for distributed NMPC using a feedback control law”. In: *Distributed Model Predictive Control Made Easy*. Springer, 2014, pp. 73–88.

[40] Michael JD Powell. “The BOBYQA algorithm for bound constrained optimization without derivatives”. In: *Cambridge NA Report NA2009/06, University of Cambridge, Cambridge* 26 (2009).

[41] Scott M Thede. “An introduction to genetic algorithms”. In: *Journal of Computing Sciences in Colleges* 20.1 (2004), pp. 115–123.

[42] V Eyert. “A comparative study on methods for convergence acceleration of iterative vector sequences”. In: *Journal of Computational Physics* 124.2 (1996), pp. 271–285.

[43] Haw-ren Fang and Yousef Saad. “Two classes of multisecant methods for nonlinear acceleration”. In: *Numerical linear algebra with applications* 16.3 (2009), pp. 197–221.

[44] Mazen Alamir. “A framework for real-time implementation of low-dimensional parameterized NMPC”. In: *Automatica* 48.1 (2012), pp. 198–204. ISSN: 0005-1098.

[45] J. Andersson et al. “CasADi: a software framework for nonlinear optimization and optimal control”. In: *Mathematical Programming Computation* 11 (2019), pp. 1–36.

[46] Boris Houska, Hans Joachim Ferreau, and Moritz Diehl. “ACADO toolkit—An open-source framework for automatic control and dynamic optimization”. In: *Optimal Control Applications and Methods* 32.3 (2011), pp. 298–312.

[47] Stefan Richter, Colin Neil Jones, and Manfred Morari. “Computational complexity certification for real-time MPC with input constraints based on the fast gradient method”. In: *IEEE Transactions on Automatic Control* 57.6 (2011), pp. 1391–1403.

[48] Ruben Van Parys et al. “Real-time proximal gradient method for embedded linear MPC”. In: *Mechatronics* 59 (2019), pp. 1–9.

[49] Jakub Novák and Petr Chalupa. “Implementation aspects of embedded MPC with fast gradient method”. In: *International Journal of Circuits, Systems and Signal Processing* (2014).

[50] Markus Kögel and Rolf Findeisen. “A fast gradient method for embedded linear predictive control”. In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 1362–1367.

[51] David Q Mayne. “Model predictive control: Recent developments and future promise”. In: *Automatica* 50.12 (2014), pp. 2967–2986.

[52] Maria M Seron, Jose A De Dona, and Graham C Goodwin. "Global analytical model predictive control with input constraints". In: *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No. 00CH37187)*. Vol. 1. IEEE. 2000, pp. 154–159.

[53] Daniel E Quevedo, Graham C Goodwin, and José A De Doná. "Finite constraint set receding horizon quadratic control". In: *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal* 14.4 (2004), pp. 355–377.

[54] Angelo D Bonzanini et al. "Toward safe dose delivery in plasma medicine using projected neural network-based fast approximate NMPC". In: *IFAC-PapersOnLine* 53.2 (2020), pp. 5279–5285.

[55] Kimberly J Chan, Joel A Paulson, and Ali Mesbah. "Deep learning-based approximate nonlinear model predictive control with offset-free tracking for embedded applications". In: *2021 American Control Conference (ACC)*. IEEE. 2021, pp. 3475–3481.

[56] Nirlipta Ranjan Mohanty et al. "Hardware Implementation of Low-complexity Deep Learning-based Model Predictive Controller". In: *2021 European Control Conference (ECC)*. IEEE. 2021, pp. 1024–1029.

[57] Saket Adhau et al. "Embedded implementation of deep learning-based linear model predictive control". In: *2019 sixth Indian control conference (ICC)*. IEEE. 2019, pp. 200–205.

[58] Mazen Alamir. "Fast NMPC: A reality-steered paradigm: Key properties of fast NMPC algorithms". In: *2014 European Control Conference (ECC)*. IEEE. 2014, pp. 2472–2477.

[59] Jonathan Barzilai and Jonathan M Borwein. "Two-point step size gradient methods". In: *IMA journal of numerical analysis* 8.1 (1988), pp. 141–148.

[60] Yu-Hong Dai, Li-Zhi Liao, and Duan Li. "On restart procedures for the conjugate gradient method". In: *Numerical Algorithms* 35.2 (2004), pp. 249–260.

[61] Ján Drgoňa et al. "Approximate model predictive building control via machine learning". In: *Applied Energy* 218 (2018), pp. 199–216.

[62] Xuan-Huy Pham et al. "Revisiting a fixed-point hierarchical control design for cryogenic refrigerators under constraints, nonlinearities and real-time considerations". In: *European Journal of Control* 63 (2022), pp. 82–96.

[63] Homer F Walker and Peng Ni. "Anderson acceleration for fixed-point iterations". In: *SIAM Journal on Numerical Analysis* 49.4 (2011), pp. 1715–1735.

[64] RR Meyer. "On the convergence of algorithms with restart". In: *SIAM Journal on Numerical Analysis* 13.5 (1976), pp. 696–704.

[65] Phanisri P Pratapa and Phanish Suryanarayana. "Restarted Pulay mixing for efficient and robust acceleration of fixed-point iterations". In: *Chemical Physics Letters* 635 (2015), pp. 69–74.

[66] Sydney Lawrence Dixon and Cesare Hall. *Fluid mechanics and thermodynamics of turbomachinery*. Butterworth-Heinemann, 2013.

[67] Fourth Edition. "Control Valve Handbook". In: (2005).

[68] François Bonne. "Modélisation et contrôle des grands réfrigérateurs cryogéniques". Theses. Université de Grenoble, Dec. 2014. URL: `https://tel.archives-ouvertes.fr/tel-01133432`.

[69] Stefano Bracco, Ilka Faccioli, and M Troil. “A numerical discretization method for the dynamic simulation of a double-pipe heat exchanger”. In: *International Journal of Energy* 1.3 (2007), pp. 47–58.

[70] MR Ansari and V Mortazavi. “Simulation of dynamical response of a countercurrent heat exchanger to inlet temperature or mass flow rate change”. In: *Applied Thermal Engineering* 26.17-18 (2006), pp. 2401–2408.

[71] Arturo Zavala-Rıo and Raúl Santiesteban-Cos. “Reliable compartmental models for double-pipe heat exchangers: An analytical study”. In: *Applied Mathematical Modelling* 31.9 (2007), pp. 1739–1752.

Résumé — Les réfrigérateurs cryogéniques des grands instruments de recherche (par exemple, le LHC au CERN et le JT60SA au Japon, ITER en France) exigent une fiabilité de plus en plus grande dans les différentes phases de fonctionnement, notamment dans les phases transitoires ou en présence de charges thermiques variables. Ces systèmes se composent de plusieurs sous-processus cryogéniques qui sont contrôlés par des régulateurs locaux (normalement des régulateurs PID). Une telle stratégie décentralisée est suffisante tant que le système reste toujours autour des points nominaux. De plus, le couplage de la dynamique des contrôleurs ainsi que l'absence d'anticipation rendent ce type de contrôle peu adapté lors des phases transitoires ou en présence de charges thermiques très variables. Dans cette thèse, un cadre de contrôle hiérarchique récemment proposé sera développé pour le réfrigérateur de 400 W à 1,8 K du CEA (France). Dans ce cadre, un coordinateur est développé pour coordonner les contrôleurs locaux afin d'optimiser la performance globale. Ce cadre est démontré en étant appliqué au contrôle du réfrigérateur de 400 W à 1,8 K, dans lequel les modèles non linéaires, les contraintes d'actionnement et la mise en œuvre en temps réel sont pris en compte. Des approches de Fast-NMPC et d'apprentissage profond sont également étudiées et mises en œuvre au niveau des contrôleurs locaux afin de réduire le temps de calcul et de rendre le cadre proposé réalisable en temps réel. Enfin, ce cadre est appliqué pour contrôler un réfrigérateur plus complexe. Pour ce faire, certains développements, qui concernent une méthode qui converge la communication entre le coordinateur et les sous-systèmes et un solveur d'optimisation pour le coordinateur, sont mis en œuvre.

Mots clés : Contrôle décentralisé, contrôle hiérarchique, MPC, NMPC, itération à point fixe, réfrigérateur cryogénique.

Abstract — The cryogenic refrigerators in the grand research instruments (e.g. LHC in CERN and JT60SA in Japan, ITER in France) require more and more reliability in the various operating phases, especially in transient phases or in the presence of variable thermal loads. Such systems consists of several cryogenic sub-processes which are controlled by the local controllers (normally PID controllers). Such decentralized strategy is sufficient as long as the system always stay around the nominal points. In addition, the coupling of the dynamics of the controllers as well as the absence of anticipation make this type of control not very suitable during transient phases or in the presence of highly variable thermal loads. In this thesis, a recently proposed hierarchical control framework will be developed for the 400 W at 1.8 K refrigerator at CEA (France). In this framework, an coordinator is developed to coordinate the local controllers in order to optimize the global performance. Such framework is demonstrated by being applied to control the 400 W at 1.8 K refrigerator, in which nonlinear models, actuator constraints and real-time implementation are considered. Fast-NMPC and deep learning approaches are also studied and implemented at the local controllers in order to reduce the computation time and make the proposed framework feasible in real-time. Finally, this framework is applied to control a more complex refrigerator. To make this happen, some developments, which concerns an method that converges the communication between the coordinator and the subsystems and an optimization solver for the coordinator, are implemented.

Keywords: Decentralized control, hierachical control, MPC, NMPC, fixed-point iteration, cryogenic refrigerator.