Commande prédictive avec Python. Application au pilotage optimal du chauffage d'un bâtiment.

Pierre Haessig, Sylvain Chatel,
Romain Bourdais, Amanda Abreu, Hervé Guéguen

CentraleSupélec – IETR

PyCon-FR, Rennes, 16 octobre 2016

http://pierreh.eu pierre.haessig@centralesupelec.fr

Plan de la présentation

Plan de la présentation

Énergie thermique dans le bâtiment

Enjeu énergétique et environnemental

Chauffer/refroidir un bâtiment consomme *beaucoup* d'énergie.

Pour réduire cette consommation, deux types de leviers :

- améliorer la **structure** (isolation, ...) du bâtiment
  $\rightarrow$ *"hardware upgrade"*

Énergie thermique dans le bâtiment

Enjeu énergétique et environnemental

Chauffer/refroidir un bâtiment consomme *beaucoup* d'énergie.

Pour réduire cette consommation, deux types de leviers :

- améliorer la **structure** (isolation, ...) du bâtiment
  $\rightarrow$ *"hardware upgrade"*
- améliorer la **commande** (pilotage) du chauffage/clim.
  $\rightarrow$ *"software upgrade"* [sujet du jour]

Maison & bâtiment connectés

Les bâtiments n'échappent pas à la vague d'équipement en moyens de calcul et de communication numérique :

- solutions clé en main fermées : Nest (filiale Google), industriels historiques de la domotique...

Maison & bâtiment connectés

Les bâtiments n'échappent pas à la vague d'équipement en moyens de calcul et de communication numérique :

- solutions clé en main fermées : Nest (filiale Google), industriels historiques de la domotique...
- envie de solutions ouvertes : "domotique libre"
  - bcp de solutions d'interfaçage (hardware & software)
  - bcp de solutions de logging, affichage, monitoring

Maison & bâtiment connectés

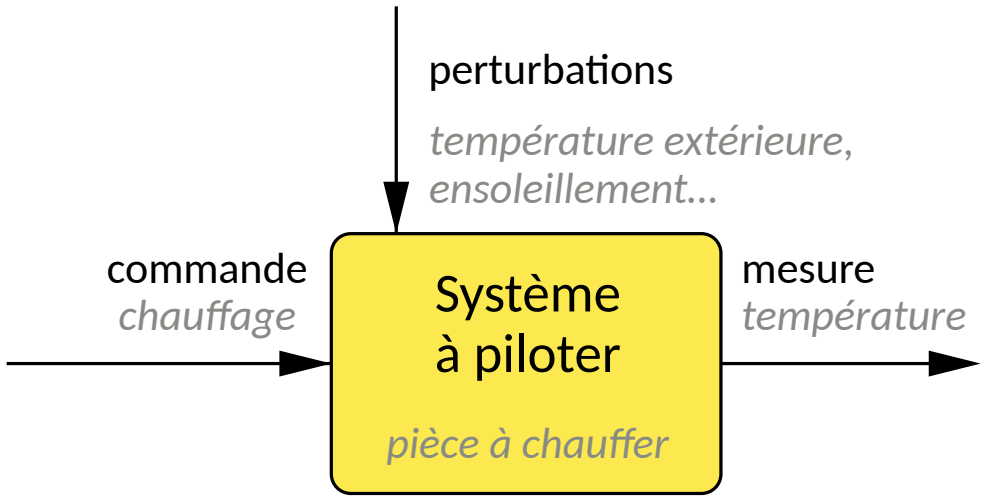
Les bâtiments n'échappent pas à la vague d'équipement en moyens de calcul et de communication numérique :

- solutions clé en main fermées : Nest (filiale Google), industriels historiques de la domotique...
- envie de solutions ouvertes : "domotique libre"
  - bcp de solutions d'interfaçage (hardware & software)
  - bcp de solutions de logging, affichage, monitoring
  - *peu de solutions de pilotage (?)* (*)

(*) je ne parle pas du pilotage par site web/appli smartphone → interfaçage

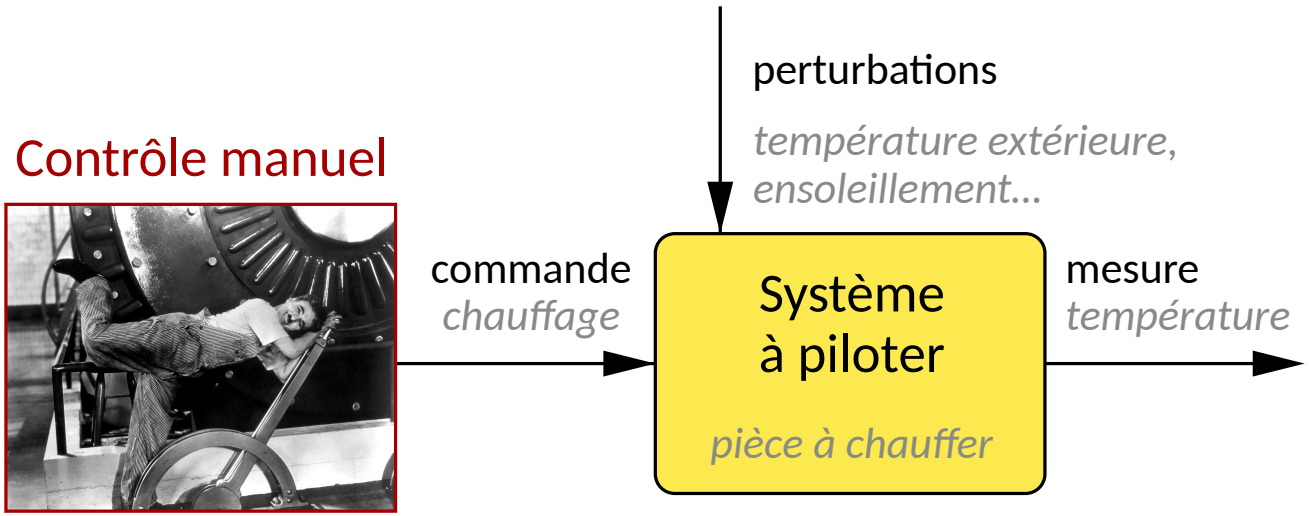
Plan de la présentation

Commande : du classique au prédictif



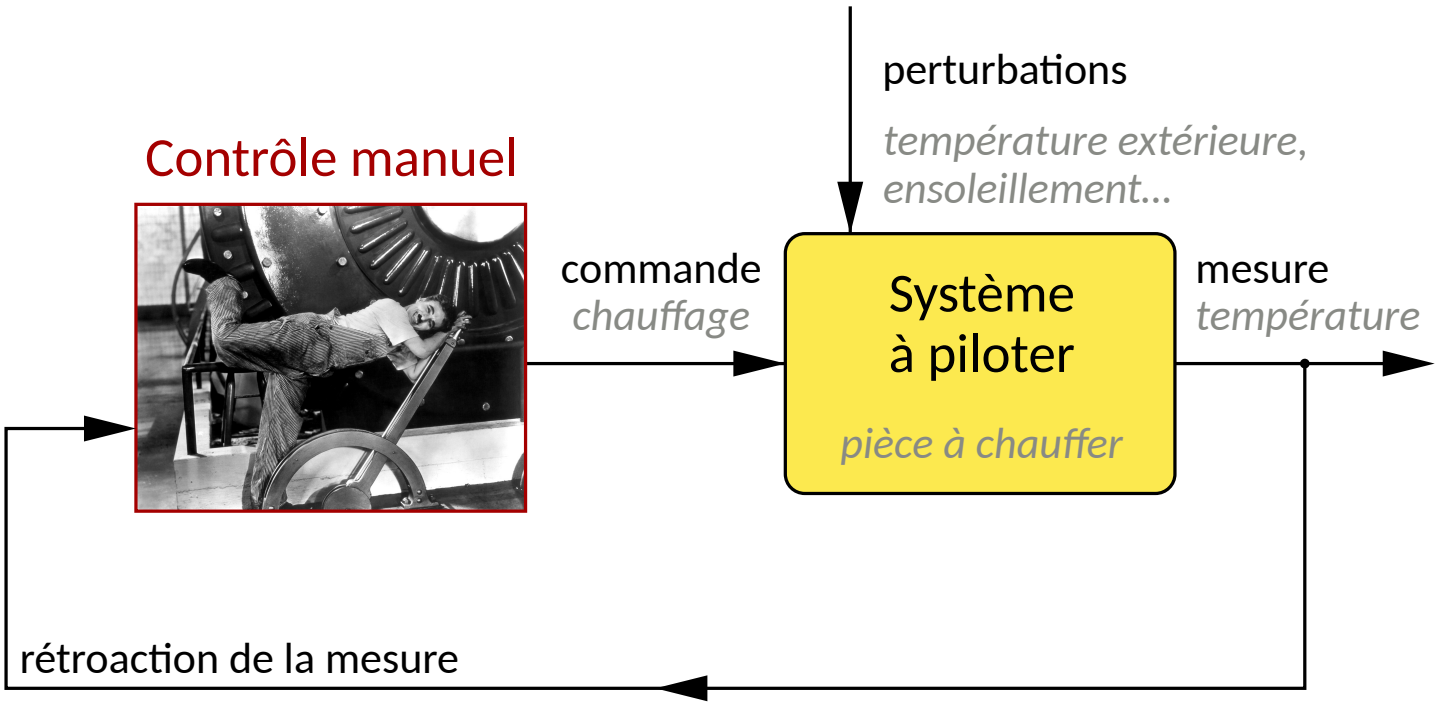
Au commencement : un système à piloter/commander

Commande : du classique au prédictif



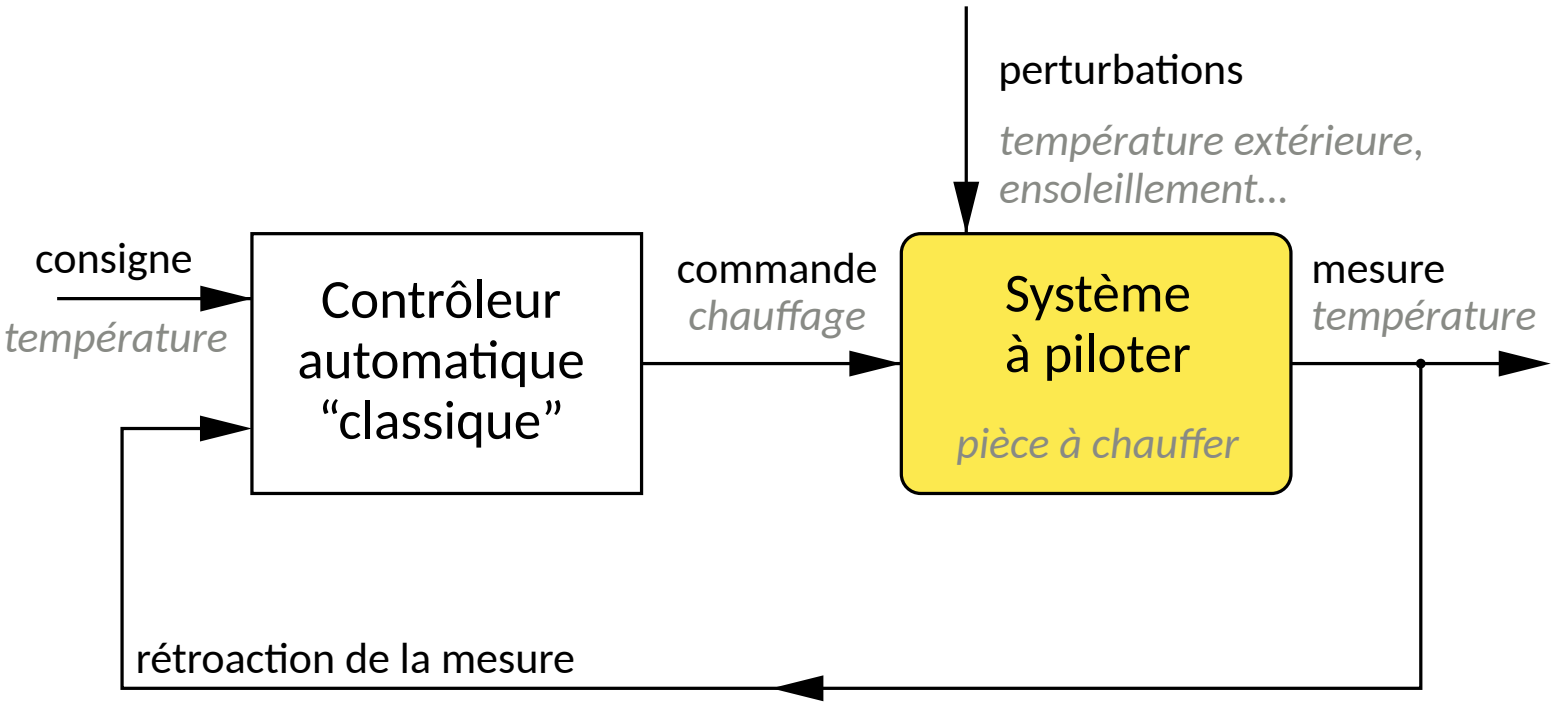
On peut piloter le système à la main...

Commande : du classique au prédictif



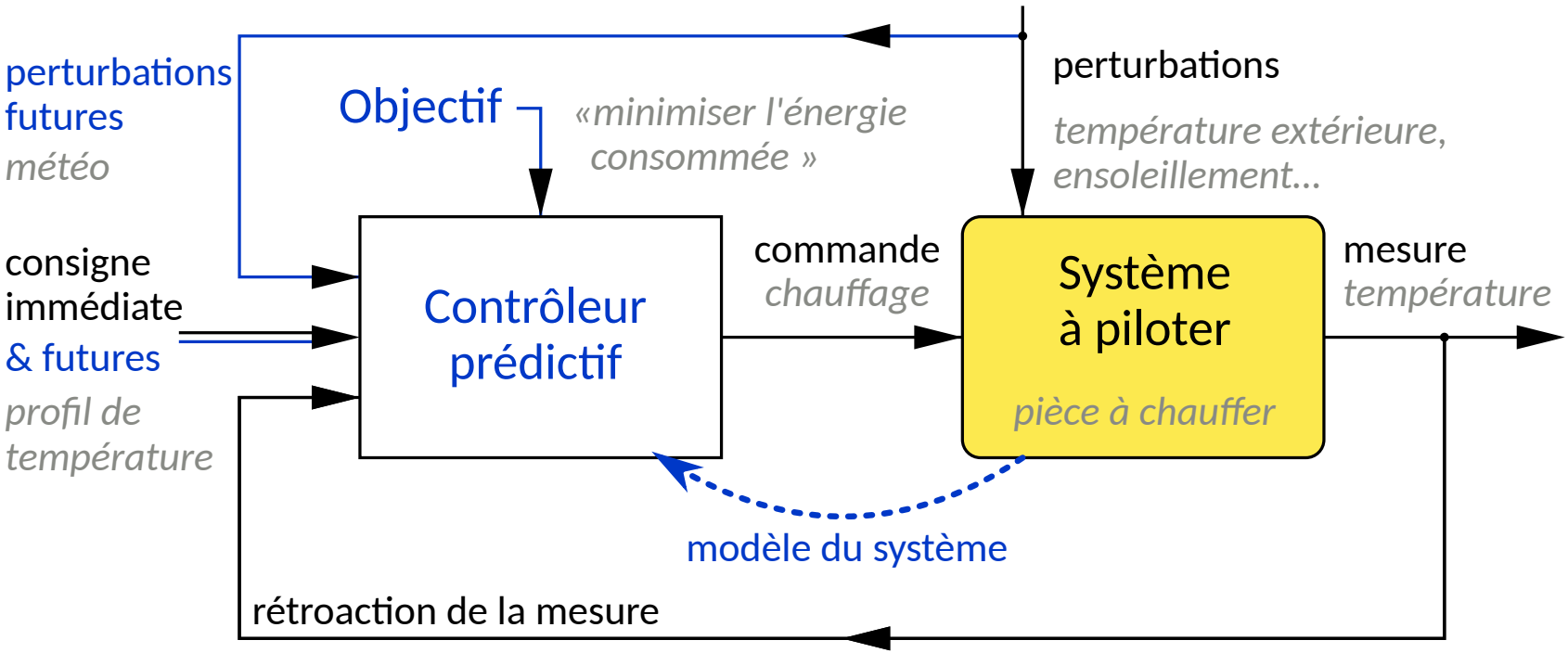
On peut piloter le système à la main...
(on s'aide généralement de la mesure
$\rightarrow$ "rétroaction", "boucle fermée")

Commande : du classique au prédictif



Commande automatique : génération de la commande pour suivre une **consigne** (PID : ~1930)

Commande : du classique au prédictif



Commande prédictive (~1970) : le simple “suivi de la consigne” est remplacé par un **objectif à optimiser**

(à minimiser ou maximiser, selon l’objectif)

Commande prédictive : les ingrédients

Le "Model Predictive Control" (MPC) se base sûr :

- **Objectif** (critère mathématique) à optimiser :
  coût économique, consommation, ...
  ou, plus classiquement, écart quadratique à une consigne,
- **Modèle de la dynamique** du système : influences des entrées commandées et extérieures (perturbations)
- **Prévisions** sur un horizon des consignes et perturbations futures

Commande prédictive : les ingrédients

Le "Model Predictive Control" (MPC) se base sûr :

- **Objectif** (critère mathématique) à optimiser :
  coût économique, consommation, ...
  ou, plus classiquement, écart quadratique à une consigne,
- **Modèle de la dynamique** du système : influences des entrées commandées et extérieures (perturbations)
- **Prévisions** sur un horizon des consignes et perturbations futures

$\rightarrow$ Travail de conception plus élevé qu'avec une "régulation classique" *(mais ça vaut le coup : très déployé industriellement)*.

Commande prédictive : besoin en calcul

Utilisation *en ligne* d'un algorithme d'optimisation :

Optimisation "en boucle fermée"

- à chaque pas, calcul d'une *séquence optimale* : commandes pour l'instant présent et les instants futurs
- seule la commande présente est effectivement appliquée
- au pas suivant, reprise du calcul sur l'horizon décalé ($\rightarrow$ « horizon glissant »)

Commande prédictive : besoin en calcul

Utilisation *en ligne* d'un algorithme d'optimisation :

Optimisation "en boucle fermée"

- à chaque pas, calcul d'une *séquence optimale* : commandes pour l'instant présent et les instants futurs
- seule la commande présente est effectivement appliquée
- au pas suivant, reprise du calcul sur l'horizon décalé ($\rightarrow$ « horizon glissant »)

$\rightarrow$ Nécessité d'embarquer de la **puissance de calcul**

Commande prédictive : besoin en calcul

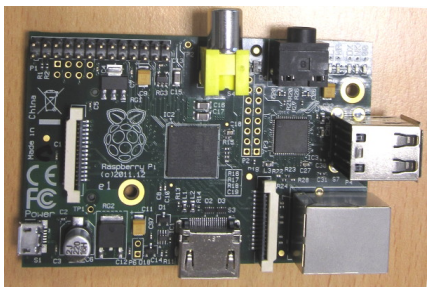
Utilisation *en ligne* d'un algorithme d'optimisation :

Optimisation "en boucle fermée"

- à chaque pas, calcul d'une *séquence optimale* : commandes pour l'instant présent et les instants futurs
- seule la commande présente est effectivement appliquée
- au pas suivant, reprise du calcul sur l'horizon décalé ($\rightarrow$ « horizon glissant »)

$\rightarrow$ Nécessité d'embarquer de la **puissance de calcul**

(mais on est en 2016)



Commande prédictive : mise en œuvre

Pour que l'optimisation *en ligne* soit :

- fiable (pas de problème de convergence)
- soluble en un temps raisonnable (complexité polynomiale)

elle est souvent formulée de façon **linéaire/quadratique**.

(ça tombe bien : "Coût $= \sum_k \text{Prix}_k \times \textbf{Puissance}_k \, \Delta_t$" est linéaire)

Commande prédictive : mise en œuvre

Pour que l'optimisation *en ligne* soit :

- fiable (pas de problème de convergence)
- soluble en un temps raisonnable (complexité polynomiale)

elle est souvent formulée de façon **linéaire/quadratique**.

(ça tombe bien : "Coût $= \sum_k \text{Prix}_k \times \textbf{Puissance}_k \, \Delta_t$" est linéaire)

$\rightarrow$ utilisation de solvers de "programmes linéaires/quadratiques" :

- commerciaux : CPLEX (IBM), Gurobi
- libres :
  - GLPK (GNU) : programmes mixtes entiers linéaires "MILP"
  - ECOS (embotech) : dédié aux systèmes embarqués
  - cvxopt : interface naturelle en Python
    (Andersen, Dahl and Vandenberghe)

Commande prédictive : en Python

NB : la R&D en commande se fait souvent sous Matlab/Simulink :

- environnement puissant, riche en fonctionnalités (e.g. automatique et traitement du signal)
- mais commercial (et sémantique du langage bof bof)

Commande prédictive : en Python

NB : la R&D en commande se fait souvent sous Matlab/Simulink :

- environnement puissant, riche en fonctionnalités (e.g. automatique et traitement du signal)
- mais commercial (et sémantique du langage bof bof)

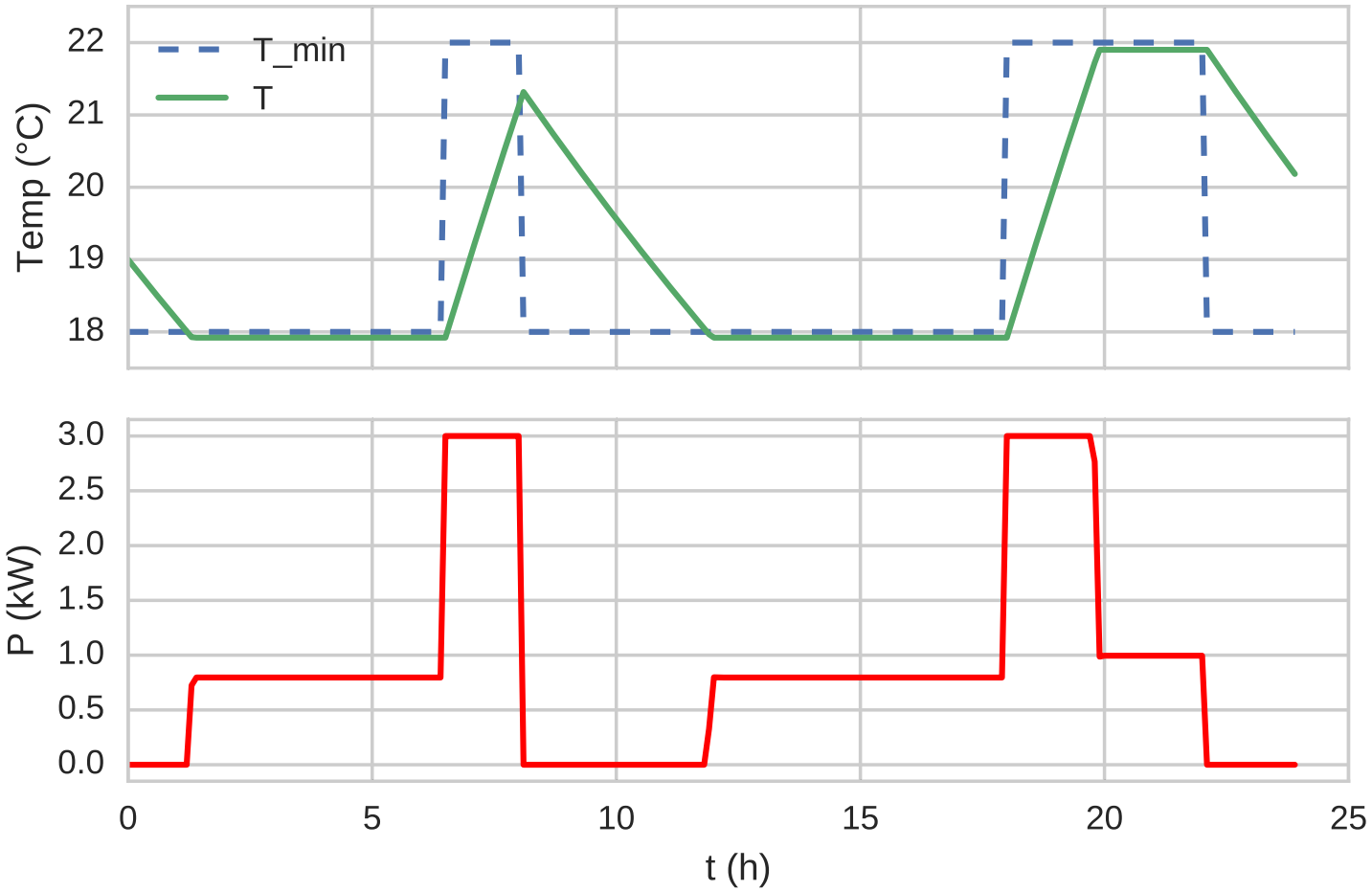
Mais on peut s'en sortir en Python :

- calcul numérique avec des matrices : **numpy**
- solver d'optimisation linéaire : **cvxopt** (nb : interfaçage aux grands solvers commerciaux possible)
- langage de modélisation : **cvxpy**
  "a Python-embedded **modeling language** for convex optimization"

`http://cvxopt.org/`, `http://cvxpy.readthedocs.io`

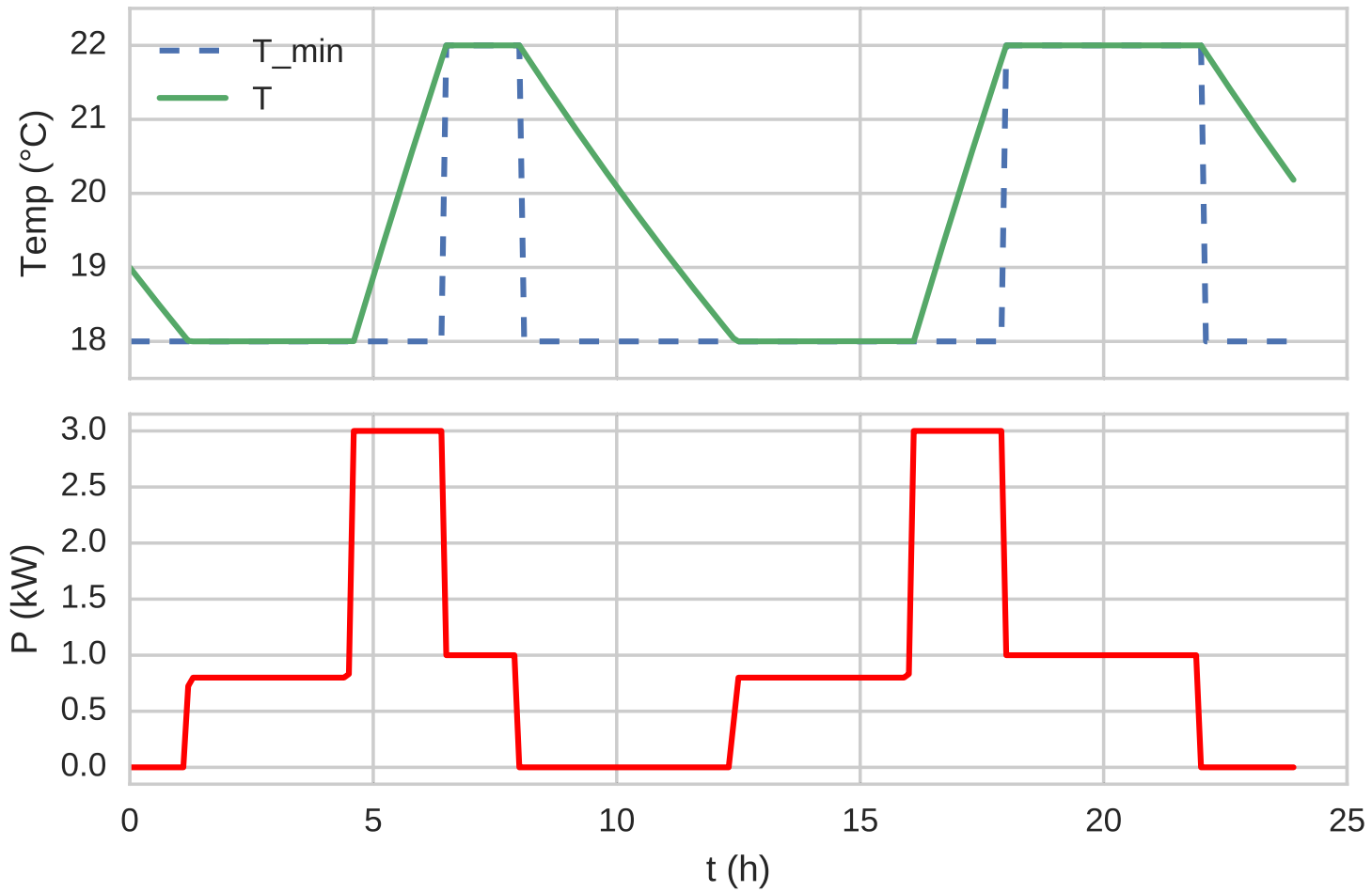
Plan de la présentation

Commande de chauffage classique



Simulation thermique sur une journée (Text = 2°C).
Régulation classique “proportionnelle”

Commande de chauffage prédictive



Simulation thermique sur une journée (Text = 2°C).
Optimisation par prog. linéaire “boucle ouverte” des 240 pas de temps.

Plan de la présentation

Conclusion

La commande prédictive :

- permet d'optimiser le pilotage d'un système
  (e.g. économie de chauffage dans un bâtiment)
- peut se mettre en œuvre en Python
  (le gros du travail est dans un solver d'optimisation)
- peut se mettre en œuvre en Python avec des logiciels libres

Conclusion

La commande prédictive :

- permet d'optimiser le pilotage d'un système
  (e.g. économie de chauffage dans un bâtiment)
- peut se mettre en œuvre en Python
  (le gros du travail est dans un solver d'optimisation)
- peut se mettre en œuvre en Python avec des logiciels libres

→ Vers de la domotique *de commande* libre ?

Perspectives

Tester sur des Raspberry Pi
($\rightarrow$ travaux à Ulm Univ., mais code source non trouvé)

(Hentzelt, Klingler, and Graichen, "Experimental Results for Distributed Model Predictive Control Applied to a Water Distribution System.", 2014)

Perspectives

Tester sur des Raspberry Pi
(→ travaux à Ulm Univ., mais code source non trouvé)
(Hentzelt, Klingler, and Graichen, "Experimental Results for Distributed Model Predictive Control Applied to a Water Distribution System.", 2014)

Pour faciliter l'utilisation du MPC : package `dmpc`

- license BSD-3 (P. Haessig & S.Chatel, 2016)
- version alpha, mais déjà `pip install`-able depuis le dépôt
- `https://github.com/pierre-haessig/python-dmpc`

Perspectives

Tester sur des Raspberry Pi
($\rightarrow$ travaux à Ulm Univ., mais code source non trouvé)
(Hentzelt, Klingler, and Graichen, "Experimental Results for Distributed Model Predictive Control Applied to a Water Distribution System.", 2014)

Pour faciliter l'utilisation du MPC : package `dmpc`

- license BSD-3 (P. Haessig & S.Chatel, 2016)
- version alpha, mais déjà `pip install`-able depuis le dépôt
- `https://github.com/pierre-haessig/python-dmpc`

télécharger diapos et notebooks :
`https://github.com/pierre-haessig/mpc-pyconfr-2016`