Consommation électrique d'un data center

Ulysse Tanguy–Bompard

Candidat n° 15937

2022-2023

Ancrage au thème et motivation

Figure : Global Security Mag
`https://www.globalsecuritymag.fr/IMG/pdf/CARTE_700x500.pdf`

- Plus de la moitié des data center français sont en zone urbaine
- 3 % de la consommation mondiale d'électricité
- Objectif : réduire leur consommation

Plan

1. Modélisation d'un data center
2. Grandeurs caractéristiques
3. Puissance consommée
4. Simulation d'un data center
5. Répartition minimisant la puissance consommée en régime stationnaire avec l'algorithme du gradient

Modélisation d'un data center

Différents modèles

- Modèle non retenu : ordinateur de bureau
  - Dangereux (230 V)
  - Grande inertie thermique
- Modèle retenu : Raspberry Pi
  - Peu dangereux
  - Réponse rapide aux perturbations

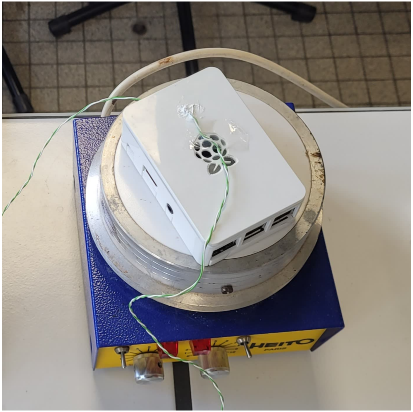
Grandeurs caractéristiques

- Masse volumique moyenne ρ :
  - Dimensions $85{,}60\,\mathrm{mm} \times 53{,}98\,\mathrm{mm} \times 17{,}00\,\mathrm{mm}$
  - Masse $90{,}9\,\mathrm{g}$
  - $\rho = 1{,}16 \times 10^3\,\mathrm{kg} \cdot \mathrm{m}^{-3}$
- Capacité thermique massique moyenne c
  - Expérience avec calorimètre
  - $c = 4{,}5\,\mathrm{kJ} \cdot \mathrm{K}^{-1} \cdot \mathrm{kg}^{-1}$
- Conductivité thermique moyenne λ

Grandeurs caractéristiques
Conductivité moyenne

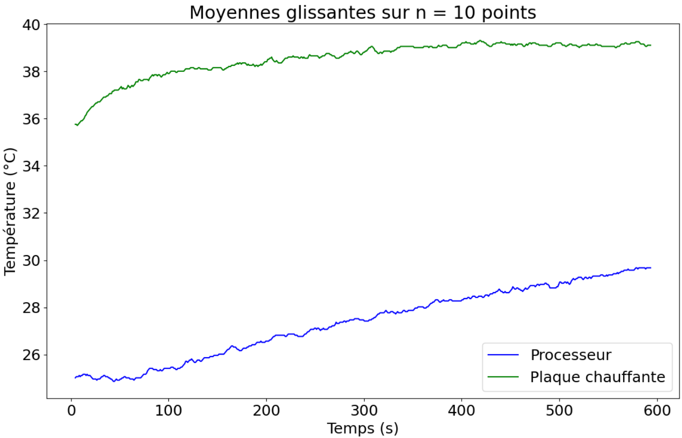
Protocole

1. Plaque chauffante
2. Mesure de la température aux deux extrémités
3. Diffusion thermique : lien entre les températures et la conductivité



Grandeurs caractéristiques

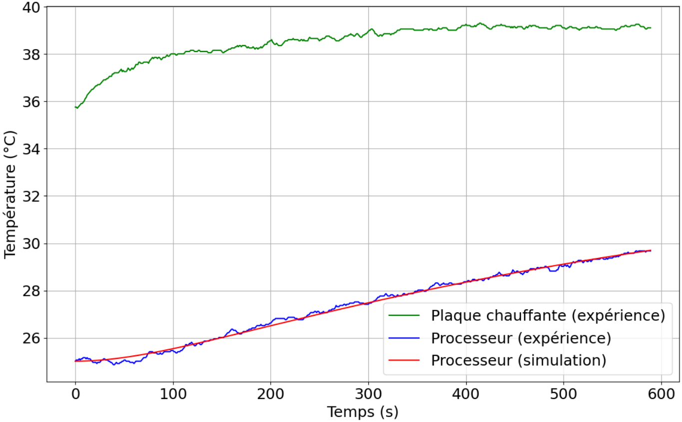
Conductivité moyenne



$$\sigma = \frac{\Delta T}{\sqrt{3}\sqrt{n}} = 0{,}09\,^{\circ}\mathrm{C}$$

Grandeurs caractéristiques

Conductivité moyenne



On détermine D grâce à l'équation de la diffusion

$D = 3{,}1 \times 10^{-5}\,\mathrm{m^2/s}$ $\qquad \lambda = D\rho c = 1{,}6 \times 10^{2}\,\mathrm{W \cdot m^{-1} \cdot K^{-1}}$

Puissance consommée

Définition de la quantité de calcul

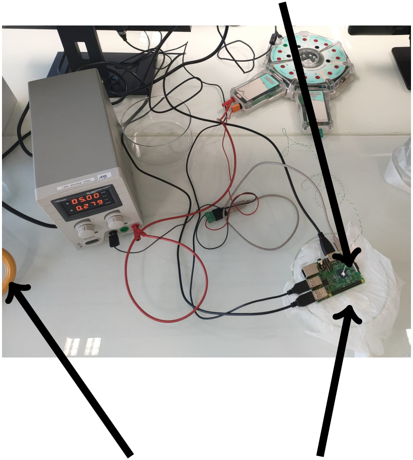
K est le nombre de calculs par seconde

```
from time import time, sleep

def calculs(n):
    a, b = 60986.5150141834, 2831540.2372984355 # arbitraires
    for i in range(n):
        c = a ** (-b)

K = 100_000 # calculs par seconde
while True:
    t_i = time()
    calculs(K)
    sleep(1 - (time() - t_i))
```

Puissance consommée
Dispositif expérimental

1. On impose les conditions extérieures en température
2. On impose la quantité de calculs
3. On attend le régime stationnaire
4. On mesure la température au processeur et l'intensité moyenne consommée

Raspberry Pi

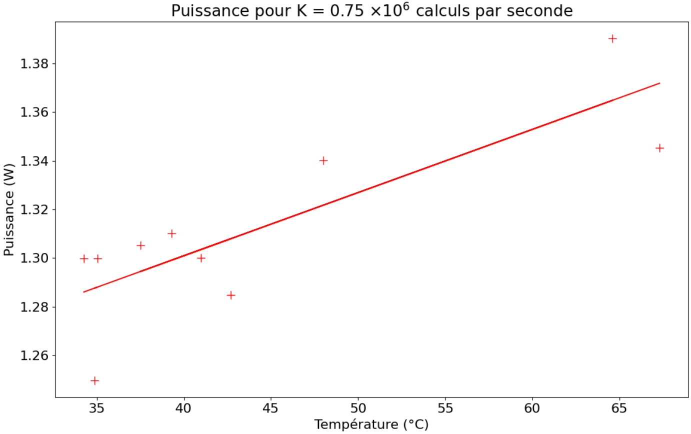


Lampe de chantier

Glace

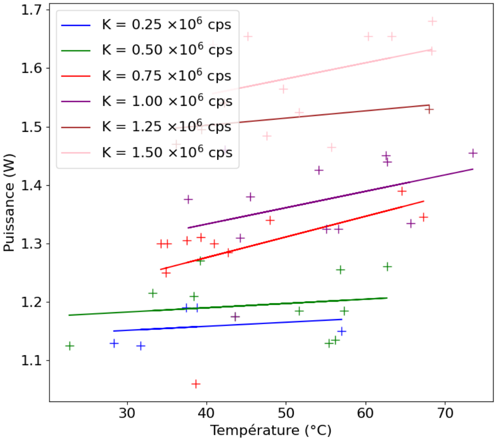
Puissance consommée

Résultats expérimentaux



Puissance consommée

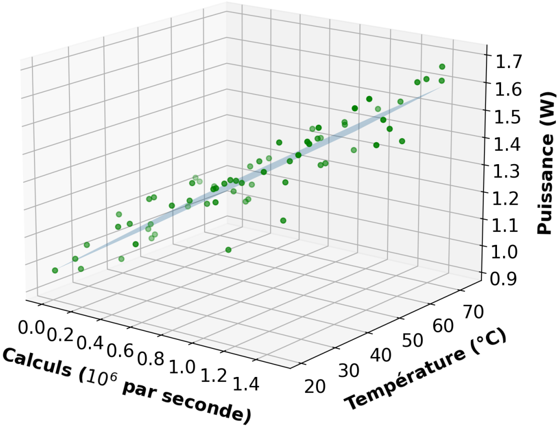
Résultats expérimentaux



On suppose qu'on peut faire un développement limité à l'ordre 1
$P = a \times T + b(K)$

Puissance consommée

Régression linéaire



Plan $P = a \times T + b \times K + c$

Moyenne des écarts relatifs : 3 %

Puissance consommée

Régression linéaire

- Allure de plan

$$P = a \times T + b \times K + c$$

- Régression linéaire avec `np.linalg.lstsq`
  - $a = 1{,}8 \times 10^{-3}\,\mathrm{W/K}$
  - $b = 3{,}6 \times 10^{-1}\,\mathrm{W}/$millions de calculs par seconde
  - $c = 0{,}94\,\mathrm{W}$

Simulation d'un data center

Hypothèses

- Carcasse de l'ordinateur : pavé uniforme
- Source thermique
  - Pavé centré sur la carcasse
  - Puissance volumique uniforme
- Air ambiant
  - On néglige la convection
  - Diffusion thermique uniquement
- Murs à température constante

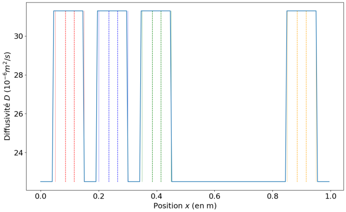
On s'attend à une convergence vers un régime stationnaire.

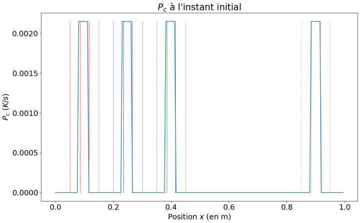
Simulation d'un data center

Principe de l'algorithme

- Temps et espace (1D) discrétisés
- Température : tableau `numpy` `T[x, t]`
- Équation aux dérivées partielles

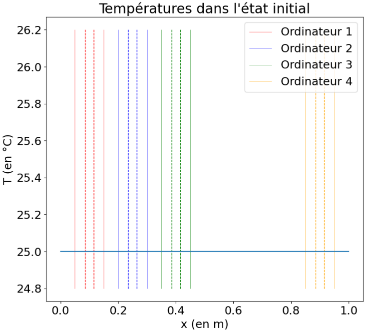
$$\frac{\partial T}{\partial t} = D(x)\frac{\partial^2 T}{\partial x^2} + P_c(x, T) \quad \text{où} \quad P_c(x, T) = \frac{P(K, T)}{C_{\text{processeur}}} \text{ ou } 0$$

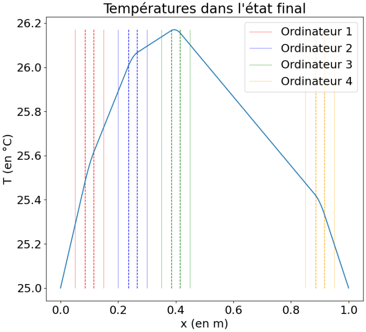




Simulation d'un data center

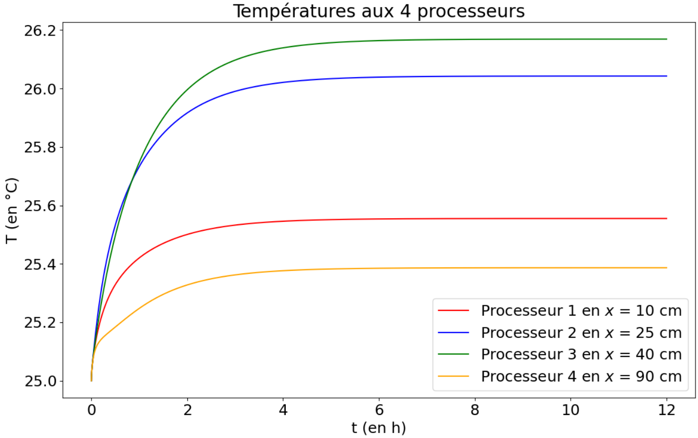
Champ de température





Simulation d'un data center

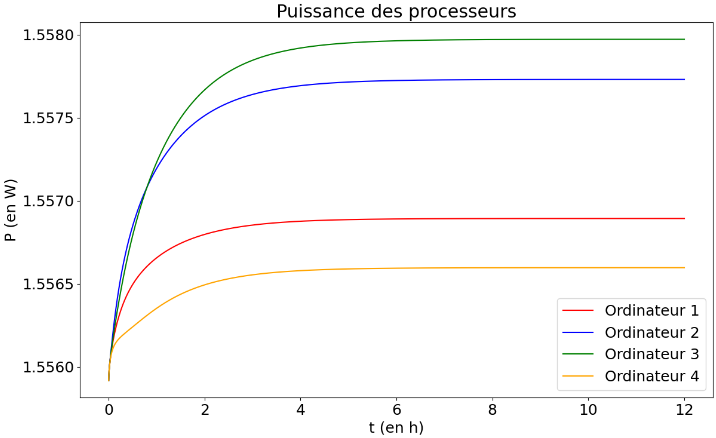
Températures aux processeurs



Régime stationnaire atteint au bout de quelques heures.

Simulation d'un data center

Puissances consommées



Régime stationnaire atteint au bout de quelques heures.

Répartition minimisant la puissance

Algorithme de descente du gradient

Ici la fonction f est la puissance consommée en régime stationnaire.

Répartition minimisant la puissance

Algorithme de descente du gradient

Ici la fonction f est la puissance consommée en régime stationnaire.

Pour trouver un minimum d'une fonction f :

- Calcul du gradient au point M, $\boldsymbol{\nabla} f(M)$
- Test d'arrêt : fin si $||\boldsymbol{\nabla} f(M)|| < \varepsilon$
- Nouveau point $M \leftarrow M - \alpha \times \boldsymbol{\nabla} f(M)$

Répartition minimisant la puissance

Algorithme de descente du gradient

Ici la fonction f est la puissance consommée en régime stationnaire.

Pour trouver un minimum d'une fonction f :

- Calcul du gradient au point M, $\nabla f(M)$
- Test d'arrêt : fin si $||\nabla f(M)|| < \varepsilon$
- Nouveau point $M \leftarrow M - \alpha \times \nabla f(M)$

On note L la longueur de la pièce et K la quantité de calculs totale.
Coordonnées réduites : $\frac{x_i}{L}$ et $\frac{K_i}{K}$ pour avoir $f : [0,1]^{2n} \rightarrow \mathbb{R}$

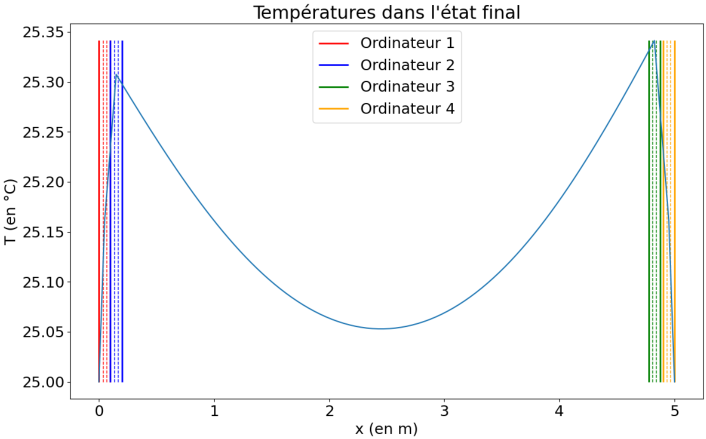
Répartition optimale

Algorithme du gradient

```
def ieme_derive_partielle(point, i):
    pnew =np.copy(point)
    pnew[i] =point[i] +pas
    return (f(pnew) -f(point)) /pas

def gradient(point):
    return np.array([
        ieme_derive_partielle(point, i)
        for i in range(len(point))])

point =point_initial
while True:
    nablaf =gradient(f, point)
    if norme(nablaf) <epsilon:
        break # sortie de l’algorithme
    point =point -pas *nablaf
print(point) # point final
```

Répartition minimisant la puissance

Minimum pour 4 ordinateurs

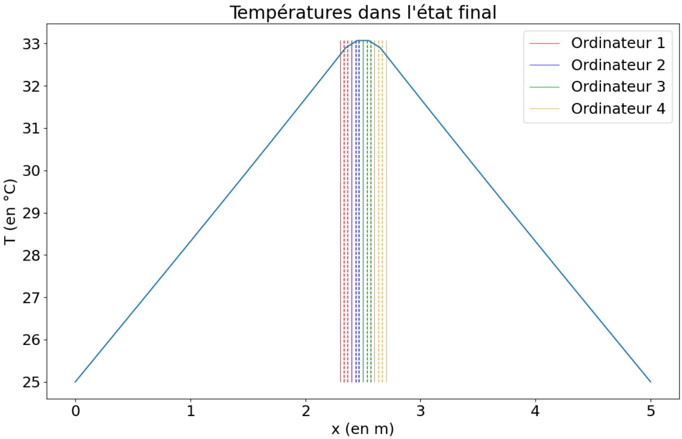


Puissance totale en régime stationnaire $P = 6{,}22\,\mathrm{W}$

Environ 10 % des calculs par ordinateur « intérieur » et 40 % par ordinateur « extérieur »

Répartition minimisant la puissance

Maximum pour 4 ordinateurs



Quasi-équirépartition des calculs

$P = 6{,}47\,\mathrm{W}$, économie de 4 %

Pour une consommation minimale :

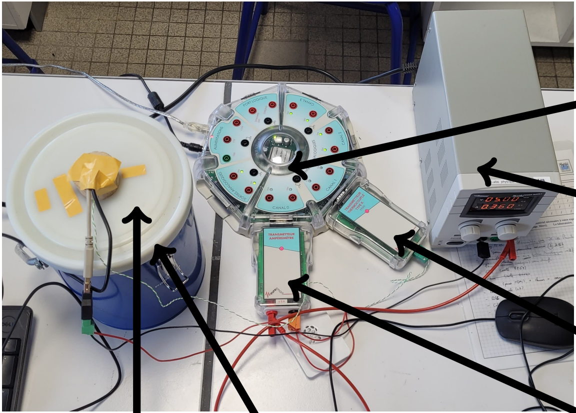
- Ne pas regrouper tous les ordinateurs
- Éloigner les ordinateurs du centre de la pièce
- Donner plus de calculs aux ordinateurs proches des murs
- Bien maintenir les murs à température constante

Conclusion

- Avantages concrets
  - Économies d'énergie de l'ordre de 4 %
  - Enjeu économique majeur
- Pistes d'approfondissement
  - Mesures plus précises des grandeurs caractéristiques
  - Simulation à 2D

Grandeurs caractéristiques

Capacité thermique



Grandeurs caractéristiques

Capacité thermique

0. Mesure de la masse en eau du calorimètre $m_{\text{calo}} = 37\,\text{g}$
1. Avant calculs (19,0 °C) intensité moyenne de 288 mA
2. Lancement des calculs à 1 min
3. Fin des calculs à 6 min. Pendant les calculs, on a
   - tension 5,0 V
   - intensité moyenne 381 mA
   - travail électrique 612 J
4. Thermalisation : sur les 100 dernières secondes, 20,6 °C

Grandeurs caractéristiques
Capacité thermique

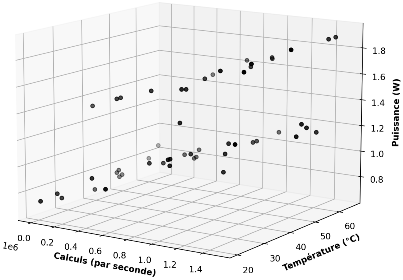
0. Mesure de la masse en eau du calorimètre $m_{\text{calo}} = 37\,\text{g}$
1. Avant calculs (19,0 °C) intensité moyenne de 288 mA
2. Lancement des calculs à 1 min
3. Fin des calculs à 6 min. Pendant les calculs, on a
   - tension 5,0 V
   - intensité moyenne 381 mA
   - travail électrique 612 J
4. Thermalisation : sur les 100 dernières secondes, 20,6 °C

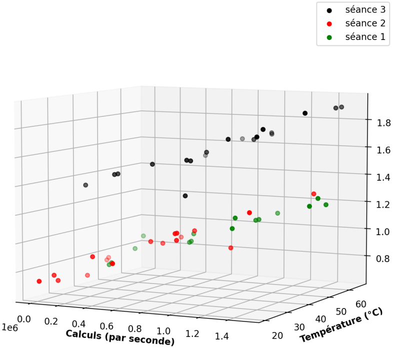
(1er principe)
$\Delta U = W = C\Delta T$

- Capacité thermique $C = 204\,\text{J/K}$
- Capacité thermique massique $c = 4{,}5\,\text{kJ} \cdot \text{K}^{-1} \cdot \text{kg}^{-1}$
- Valeur surévaluée car on a négligé les pertes

Mesures de la puissance

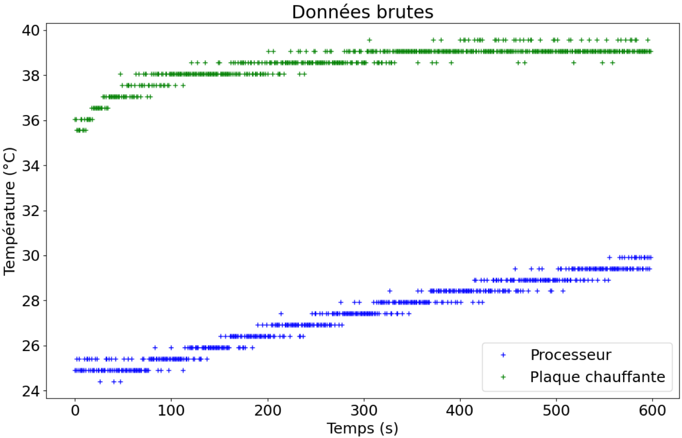
Erreurs de mesure





Conductivité moyenne

Données brutes



Conductivité moyenne

Écart-type par Monte-Carlo