Projet : Simulation numérique d'un dispositif de refroidissement

Ce projet a pour but de réaliser un programme C++ permettant d'étudier le comportement thermique d'un dispositif de refroidissement d'un micro-processeur. Un des moyens généralement utilisé pour réguler la température d'un processeur est l'utilisation d'un ventilateur. Le soufflage d'air va permettre d'évacuer la chaleur par convection sur la surface du processeur. Pour améliorer ce processus, le processeur est généralement attaché à un dissipateur (composé de plusieurs ailettes) qui est un élement très conducteur de chaleur. Ce disposif supplémentaire, représenté avec la figure 1, permet d'augmenter la surface d'échange avec le flux d'air et d'ainsi refroidir plus efficacement le composant électronique.

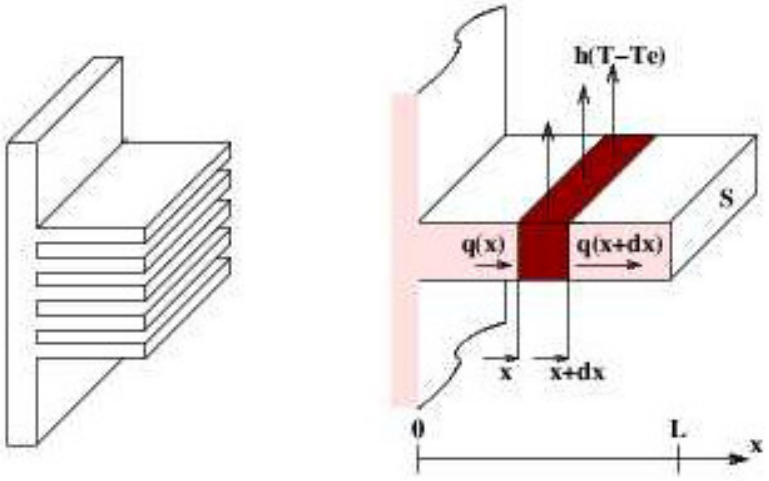


FIGURE 1 – Modélisation d'une ailette du dissipateur

Pour ce projet, nous allons nous intéresser uniquememt à la simulation thermique d'une seule ailette du dissipateur. Les données du problème sont : la géométrie de l'ailette, figure 2, qui est décrite par les longueurs L_x, L_y et L_z,le flux de chaleur Φ_p généré par le processeur et la température ambiante T_e. De plus, nous supposons que l'ailette est suffisamment mince pour considérer le problème unidimensionnel. La température T en un point donné dépendra uniquement de sa position selon x et de l'instant t.

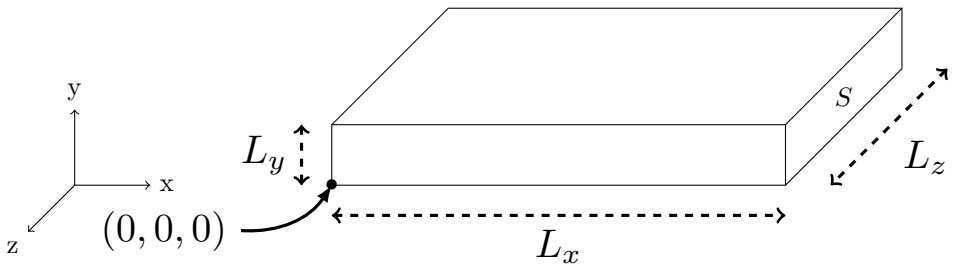


FIGURE 2 – Géométrie de l'ailette

En prenant en compte les pertes latérales par convection avec l'air, l'équation de la chaleur dans une ailette définie sur le domaine $x = [0, L_x]$ est donnée par :

$$\rho C_p \frac{\partial T}{\partial t} - \kappa \frac{\partial^2 T}{\partial x^2} + \frac{h_c\ p}{S}(T - T_e) = 0 \quad (1)$$

avec ρ la densité, C_p la chaleur spécifique à pression constante, κ la conductivité thermique, h_c le coefficent de transfert de chaleur surfacique, $S = L_y L_z$ l'aire d'une section transversale et $p = 2(L_x + L_y)$ le périmètre d'une section transversale.Au niveau des conditions aux limites, nous supposons connaître le flux de chaleur Φ_p dégagé par le processeur en $x = 0$ et nous faisons l'hyphothèse que le transfert de chaleur en $x = L_x$ est négligeable par rapport au flux apporté sur la section longitudinale. Cette modélisation nous conduit finalement à la recherche de la température T définie sur le domaine $x = [0, L_x]$ qui vérifie :

$$\rho C_p \frac{\partial T}{\partial t} - \kappa \frac{\partial^2 T}{\partial x^2} + \frac{h_c\ p}{S}(T - T_e) = 0 \quad (2)$$

$$-\kappa \left.\frac{\partial T}{\partial x}\right|_{x=0} = \Phi_p \quad (3)$$

$$-\kappa \left.\frac{\partial T}{\partial x}\right|_{x=L_x} = 0 \quad (4)$$

1 Paramètres physiques et géométriques

Les valeurs des paramètres utilisés pour les simulations sont données dans la table 1. On a choisi les propriétés physiques en considérant que l'ailette est constituée d'un alliage d'aluminium.

nom	valeur	unité	nom	valeur	unité
ρ	2700	$kg/(m^3)$	h_c	200	$W/(m^2 \cdot K)$
C_p	940	$J/(kg \cdot K)$	L_x	0.04	m
κ	164	$W/(m \cdot K)$	L_y	0.004	m
T_e	20	°C	L_z	0.05	m
Φ_p	$1.25 \cdot 10^5$	W/m^2			

TABLE 1 – Valeurs des paramètres géométriques et physiques

Cependant, les valeurs des paramètres pourront être modifiées dans le but d'analyser l'impact du paramètre sur le résultat. De plus, la valeur du coefficent de transfert de chaleur surfacique h_c est donnée dans ce tableau pour le cas où le ventilateur est en marche. On pourra également traiter le cas où le ventilateur est éteint en utilisant $h_c = 10\ W/(m^2 \cdot K)$.

2 Modèle stationnaire

Nous commençons par traiter le cas stationnaire, c'est-à-dire par calculer la distribution de température lorsque le temps t tend vers l'infini. On enlevera alors la dérivée en temps présente dans les équations 1 et 2.

Pour résoudre cette EDP, nous allons utiliser la métode des différences finies. Pour cela nous décomposons le segment $[0, L_x]$ en M intervalles de longueur $h = L_x/M$. Nous obtenons ainsi un maillage composé de M+1 points d'abscisses $x_i = i\ h, i = 0, ..., M$. La solution numérique sera alors décrite par ces M+1 points et on notera T_i la température calculée au point x_i.

Au niveau de la discrétisation de l'EDP, on utilise un développement de Taylor, ce qui permet de discrétiser la dérivée seconde à l'aide du schéma centré suivant :

$$\frac{\partial^2 T}{\partial x^2}(x_i) \approx \frac{T_{i-1} - 2T_i + T_{i+1}}{h^2}$$

Nous obtenons ainsi les équations discrètes du problème définies sur tous les noeuds internes :

$$-\kappa \frac{T_{i-1} - 2T_i + T_{i+1}}{h^2} + \frac{h_c\ p}{S}\left(T_i - T_e\right) = 0\ , \quad \forall i \in [1, M-1]$$

Reste à appliquer les conditions aux limites pour fermer le système :

$$-\kappa \left.\frac{\partial T}{\partial x}\right|_{x=0} \approx -\kappa \frac{T_1 - T_0}{h} = \Phi_p \tag{5}$$

$$-\kappa \left.\frac{\partial T}{\partial x}\right|_{x=L_x} \approx -\kappa \frac{T_{M-1} - T_M}{h} = 0 \tag{6}$$

En combinant toutes ces informations, le problème discret est décrit par la résolution d'un système $AX = F$ avec A une matrice tridiagonale (taille $M \times M$), F le vecteur second membre (taille M) et X le vecteur solution représentant la température en chaque point de discrétisation $x_i, i \in [0, M]$.

$$\underbrace{\begin{pmatrix} b_0 & c_0 & 0 & 0 & 0 & \cdots & 0 \\ a_1 & b_1 & c_1 & 0 & 0 & \cdots & 0 \\ 0 & a_2 & b_2 & c_2 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & a_{M-1} & b_{M-1} & c_{M-1} \\ 0 & \cdots & \cdots & 0 & 0 & a_M & b_M \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} T_0 \\ T_1 \\ T_2 \\ \vdots \\ \vdots \\ T_{M-1} \\ T_M \end{pmatrix}}_{X} = \underbrace{\begin{pmatrix} F_0 \\ F_1 \\ F_2 \\ \vdots \\ \vdots \\ F_{M-1} \\ F_M \end{pmatrix}}_{F}$$

avec :

- $a_i = -\frac{\kappa}{h^2} \quad \forall i \in [1, M-1]$
- $a_M = -\frac{\kappa}{h}$
- $b_i = \frac{2\kappa}{h^2} + \frac{h_c\ p}{S} \quad \forall i \in [1, M-1]$
- $b_0 = \frac{\kappa}{h}$
- $b_M = \frac{\kappa}{h}$
- $c_i = -\frac{\kappa}{h^2} \quad \forall i \in [1, M-1]$
- $c_0 = -\frac{\kappa}{h}$
- $F_i = \frac{h_c\ p}{S} T_e \quad \forall i \in [1, M-1]$
- $F_0 = \Phi_p$
- $F_M = 0$

2.1 Résolution du système algébrique

Les matrices tridiagonales peuvent être facilement inversées par la méthode de Gauss. L'idée consiste à décomposer la matrice A sous la forme $A = LU$, avec :

$$L = \begin{pmatrix} b_0^* & 0 & 0 & 0 & \cdots & 0 \\ a_1 & b_1^* & 0 & 0 & \cdots & 0 \\ 0 & a_2 & b_2^* & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{M-1} & b_{M-1}^* & 0 \\ 0 & \cdots & 0 & 0 & a_M & b_M^* \end{pmatrix}, \quad U = \begin{pmatrix} 1 & c_0^* & 0 & 0 & \cdots & 0 \\ 0 & 1 & c_1^* & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & c_{M-2}^* & 0 \\ 0 & \cdots & \cdots & 0 & 1 & c_{M-1}^* \\ 0 & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix}$$

Les coefficients b^* et c^* sont déterminés de la manière suivante :

1. $b_0^* = b_0$ et $c_0^* = c_0 / b_0^*$
2. pour k allant de 1 jusqu'à $M-1$:
   - $b_k^* = b_k - a_k \cdot c_{k-1}^*$
   - $c_k^* = c_k / b_k^*$
3. $b_M^* = b_M - a_M \cdot c_{M-1}^*$

Une fois les matrices triangulaires L et U identifiées, la résolution du système $AX = F$ s'effectue en 2 étapes :

$$AX = F \implies L\underbrace{UX}_{Y} = F \implies$$

1. Résoudre $LY = F$
   - $Y_0 = F_0/b_0^*$
   - $Y_k = (F_k - a_k \cdot Y_{k-1})/b_k^*, \quad k = 1, ..., M$
2. Résoudre $UX = Y$
   - $X_M = Y_M$
   - $X_k = Y_k - c_k^* \cdot X_{k+1}, \quad k = M-1, ..., 0$

2.2 Analyse des résultats

Une solution numérique pourra être exportée dans un fichier de manière à pouvoir visualiser cette solution sous la forme d'un graphique. Nous aurons la position x en abscisse et la valeur de la température en ordonnée. Pour cela, nous utiliserons le format CSV (Comma-separated values), qui est un format informatique ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules. Les données dans le fichier sont représentées sous forme de colonne. La première ligne correspond au nom de chaque donnée. Un exemple de ce type de fichier est donné dans la table 2 ainsi que sa représentation tabulaire et graphique. Il existe de très nombreux outils pour visualiser ce type de format : Plotly, octave/matlab, latex (Pgfplots), ParaView, ...

Fichier au format .csv	Représentation tabulaire	Représentation graphique

```
x, sol1, sol2
0, 2.3, 3.4
0.5, 3.2, 6.8
1, 2.7, 5.3
1.5, 4.1, 2.4
2, 4.4, 2.2
```

x	sol1	sol2
0	2.3	3.4
0.5	3.2	6.8
1	2.7	5.3
1.5	4.1	2.4
2	4.4	2.2

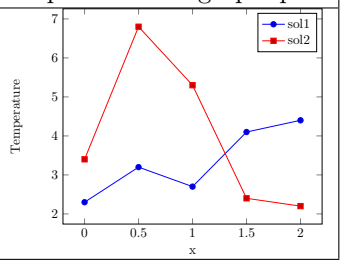


TABLE 2 – Exemple de fichier au format csv

De plus, le problème admet une solution analytique, notée T_{exact}, qui est décrite par l'expression suivante :

$$T_{\text{exact}} = T_e + \frac{\Phi_p}{\kappa} \frac{\cosh\left(\sqrt{a} L_x\right)}{\sqrt{a}\sinh\left(\sqrt{a} L_x\right)} \frac{\cosh\left(\sqrt{a}\left(L_x - x\right)\right)}{\cosh\left(\sqrt{a} L_x\right)} \quad \text{avec } a = \frac{h_c\ p}{\kappa\ S} \tag{7}$$

Cette solution exacte va nous permettre de vérifier l'implémentation de la méthode numérique. Elle pourra également être ajoutée aux fichier .csv (dans une colonne supplémentaire), dans le but de faire une comparaison visuelle des solutions. Les figures 3a et 3b illustrent ce type de résultat.

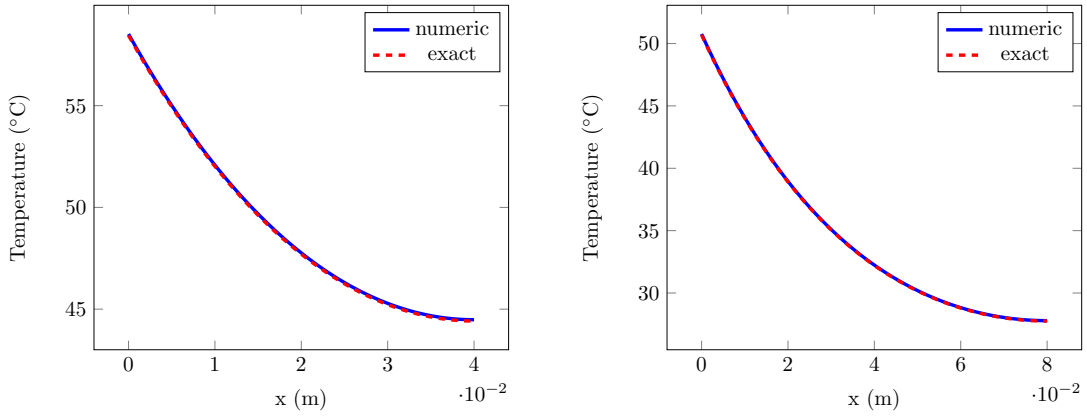
3 Modèle instationnaire

Nous souhaitons maintenant suivre l'évolution en temps de la distribution de température dans l'ailette pendant une durée notée t_{final}. Pour cela, nous devons prendre en compte la dérivée en temps dans l'équation aux dérivées partielles.

Nous commençons par échantillonnner l'intervalle de temps $[0, t_{\text{final}}]$ avec une discrétisation régulière composée de N+1 temps discret. Le pas de temps (i.e. la durée entre 2 temps discrets), sera donc constant et noté Δt. Ainsi, l'échantillonnage en temps est décrit par l'ensemble $\{t_n = n\Delta t, n = 0, ..., N\}$. On utilisera pour cette étude $t_{\text{final}} = 300s$ et $N = 600$. On note T_i^n la température discrète calculée au points x_i et à l'instant t_n.

La dérivée en temps est discrétisée à l'aide d'un schéma d'Euler :

$$\frac{\partial T}{\partial t}\left(x_i, t_n\right) \approx \frac{T_i^{n+1} - T_i^n}{\Delta t}$$



(a) $L_x = 40\ mm$ (b) $L_x = 80\ mm$

FIGURE 3 – Comparaison entre la solution numérique et la solution exacte du modèle stationnaire.

Nous obtenons ainsi les équations discrètes du problème instationnaire définies sur tous les noeuds internes à chaque temps $t_n, n = 1, ..., N$:

$$\rho C_p \frac{T_i^{n+1} - T_i^n}{\Delta t} - \kappa \frac{T_{i-1}^{n+1} - 2T_i^{n+1} + T_{i+1}^{n+1}}{h^2} + \frac{h_c\ p}{S}\left(T_i^{n+1} - T_e\right) = 0\ , \quad \forall i \in [1, M-1] \tag{8}$$

et les conditions aux limites :

$$-\kappa \left.\frac{\partial T}{\partial x}\right|_{x=0} \approx -\kappa \frac{T_0^{n+1} - T_1^{n+1}}{h} = \Phi_p \tag{9}$$

$$-\kappa \left.\frac{\partial T}{\partial x}\right|_{x=L_x} \approx -\kappa \frac{T_M^{n+1} - T_{M-1}^{n+1}}{h} = 0 \tag{10}$$

De plus, il est nécessaire de définir une solution initiale à t_0. Nous choisissons de considérer que cette solution initiale est égale à la température ambiante, c'est-à-dire que $T_i^0 = T_e, \forall i = 0, ..., M$.

Algorithm 1 Résolution du modèle instationnaire

initialiser T^0
for n=1 à N **do**
  calculer T^{n+1} en fonction de T^n
  $T^n \leftarrow T^{n+1}$
end for

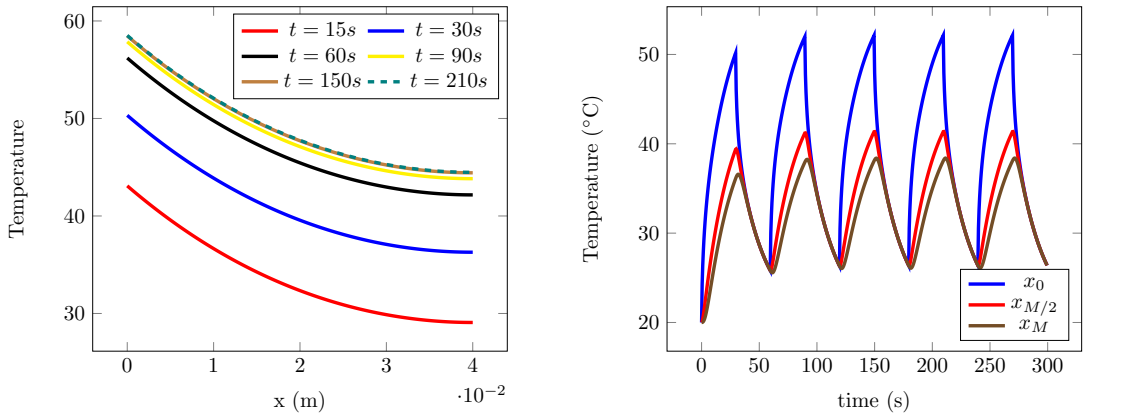
Au niveau du flux de chaleur Φ_p, deux scénarios seront envisagés. Le premier consistera à utiliser une valeur constante $\Phi_p = 1.25 \cdot 10^5$ comme pour le cas stationnaire. L'autre scénario consistera à activer et désactiver le flux toutes les 30 secondes :

1. $\Phi_p = 1.25 \cdot 10^5$ de 0 à 30 s
2. $\Phi_p = 0$ de 30 à 60s
3. $\Phi_p = 1.25 \cdot 10^5$ de 60 à 90 s
4. $\Phi_p = 0$ de 90 à 120s
5. $\Phi_p = 1.25 \cdot 10^5$ de 120 à 150 s
6. etc...

3.1 Analyse des résultats

Comme pour le cas stationnaire, il sera intéressant de visualiser la solution numérique 1d à l'aide du format csv. On pourra afficher quelques solutions en temps pour voir son évolution. On

pourra également tracer l'évolution thermique en temps de certains points de l'ailette, comme par exemple pour $x = x_0$,$x = x_{M/2}$ et $x = x_M$. Ce fichier csv sera alors composé de 4 colonnes, une première pour le temps, et les 3 autres pour la mesure de la température en ces 3 points.



(a) Flux de chaleur constant

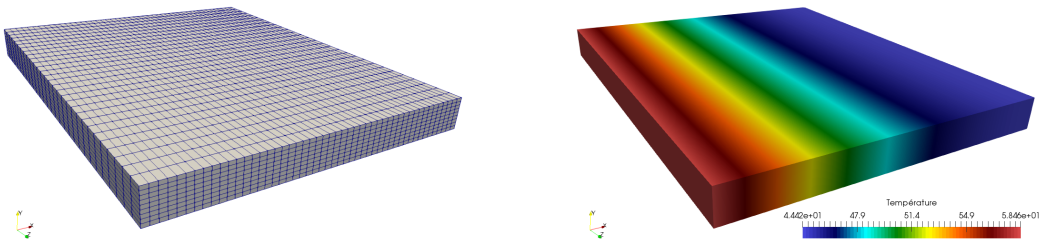
(b) Avec activation/désactivation du flux de chaleur

FIGURE 4 – Exemples de résultats pour le modèle instationnaire.

4 Visualisation 3d de la solution

Nous souhaitons maintenant visualiser la solution sur la géométrie 3d de l'ailette, figure 2. C'est un pavé droit paramétrisé par les trois longueurs L_x, L_y et L_z. Nous commençons par discrétiser les intervalles $[0, L_x]$, $[0, L_y]$ et $[0, L_z]$ en $M_x + 1$, $M_y + 1$ et $M_z + 1$ points respectivement. Les paramètres M_x, M_y et M_z seront définis par l'utilisateur et devront être strictement supérieur à 0. Attention, le paramètre M_x ne sera pas forcément le même que le paramètre M utilisé dans la discrétisation du problème 1d.

En effectuant le produit cartésien de ces 3 discrétisations d'intervalle, nous obtenons une discrétisation de la géométrie volumique (l'ailette) composée de $(M_X + 1)(M_y + 1)(M_z + 1)$ points. Cela nous permet de construire un maillage structuré formé d'un ensemble de pavé droit. Un exemple de maillage de l'ailette est illustré avec la figure 5a. On note $P_{ijk} = (x_{ijk}, y_{ijk}, z_{ijk})$ un point de la discrétisation volumique où i,j et k représentent respectivement l'indice dans la discrétisation de $[0, L_x]$, $[0, L_y]$ et $[0, L_z]$



(a) Maillage 3d ($M_x = 50, M_y = 10, M_z = 30$)

(b) Température calculée sur le maillage 3d

FIGURE 5 – Exemples de visualisation 3d.

Il nous faut maintenant définir la valeur de la température à chaque P_{ijk}, noté T_{ijk}, à partir de la solution numérique $T_i, i = 0, ..., M$ calculée sur $[0, L_x]$. On exprime la température dans le volume par :

$$T_{ijk} = \hat{T}_i, \quad \forall i = 0, ..., M_x, \forall j = 0, ..., M_y, \forall k = 0, ..., M_z \tag{11}$$

avec $\hat{T}_i, i = 0, ..., M_x$ un interpolant linéaire de la solution numérique 1d au point x_{ijk} sur le maillage volumique. Cet interpolant dépend uniquement de la position en x car $x_{i00} = x_{ijk}\ \forall j = 0, ..., M_y, \forall k = 0, ..., M_z$. La figure 5b montre la visualisation du champ de température interpolé sur le maillage 3d.

La construction de $\hat{T}$ pourra être réalisée par l'algorithme 2.

Algorithm 2 Calcul de $\hat{T}_i$

for i=0 à M_x **do**
  localiser x_{i00} dans le maillage 1d (i.e. trouver k tel que $x_{i00} \in [x_k, x_{k+1}]$)
  calculer les coefficients de la droite y=ax+b passant par les points (x_k, T_k) et (x_{k+1}, T_{k+1})
  évaluer $\hat{T}_i = a\ x_{i00} + b$
end for

Nous allons maintenant sauvegarder les données P_{ijk} et T_{ijk} dans un fichier en utilisant le format vtk. Ce fichier pourra ensuite être lu par le logiciel de visualisation ParaView qui est disponible sur toutes les plateformes (https://www.paraview.org/). Un exemple très simple de fichier vtk (exemple.vtk) est fourni avec le projet. La structure du fichier est la suivante :

On commence par l'en-tête qui initilialise le format et définit le type de maillage, ici maillage structuré :

```
# vtk DataFile Version 2.0
vtk output
ASCII
DATASET STRUCTURED_GRID
```

Ensuite, nous avons les dimensions de la discrétisation (i.e. $M_x = 3, M_y = 4, M_z = 2$) :

```
DIMENSIONS 3 4 2
```

On déclare maintenant le nombre de point $(M_x \cdot M_y \cdot M_z)$ avec la précision numérique (float)

```
POINTS 24 float
```

On trouve après la valeur de chaque point P_{ijk} décrite par ces 3 coordonnées en faisant d'abord varier i, puis j et enfin k.

```
0 0 0
1 0 0
2 0 0
0 1 0
1 1 0
2 1 0
0 2 0
...
1 2 1
2 2 1
0 3 1
1 3 1
2 3 1
```

Avec ces informations, il est déjà possible de lire ce fichier avec ParaView pour visualiser le maillage.

Il reste maintenant à rentrer les données de la solution numérique. On commence par une section FIELD avec le type de donnée, ici FieldData. Le 1 correspond au fait que l'on va exporter un seul champ (solution). S'il y en avait plusieurs, on remplacerait le 1 par le nombre de champ.

```
FIELD FieldData 1
```

Ensuite on décrit le champ avec son nom (sol1), le nombre de composantes (1 pour champ scalaire), le nombre de points (identique que pour le maillage) et la précision numérique (float)

```
sol1 1 24 float
```

Enfin, on met la valeur du champ pour chaque point en respectant l'ordre qui a été utilisé précédemment avec le maillage.

```
0
1
2
...
21
22
23
```

Pour les simulations en temps, on sauvegardera un fichier par pas de temps. Tous ces fichiers devront se trouver dans le même dossier et leurs noms devront intégrer un compteur de la forme suivante :

masolution.0.vtk masolution.1.vtk masolution.2.vtk ...

ParaView sera alors capable de lire tous ces fichiers d'un seul coup en proposant d'ouvrir le fichier masolution...vtk

5 Objectifs du projet

Le projet a pour but de construire un programme C++ permettant de calculer les solutions numériques du modèle stationnaire et instationnaire. Ces solutions seront ensuite exportées dans des fichiers aux formats csv et vtk. Le code devra être programmé en se souciant des performances. Une simulation du modèle stationnaire sur une dizaine de milliers de points de discrétisation devra être exécutée en quelques secondes. Il faudra également penser à bien structurer son programme, éviter la duplication de code et le rendre lisible et réutilisable.

L'utilisateur devra également pouvoir paramétrer le modèle en utilisant un fichier de configuration. Il contiendra les valeurs des paramètres L_x, L_y, L_z. Le fichier de configuration sera lu par le progamme et il permettra d'initialiser la valeur des paramètres utilisés dans la simulation. Voici un exemple d'un fichier de configuration :

simu.cfg

```
Lx 40 Ly 4 Lz 50
M 10000
Phi 0.125
hc 0.0002
Te 20
stationary 0
TFinal 300
N 600
Mx 50 My 10 Mz 30
```

Pour que le programme puisse lire ce fichier de configuration, le nom du fichier sera donné en option à l'éxécutable :

```
user $ ./myprog simu.cfg
```

Un rapport sera accompagné du programme C++. Il devra contenir une description de la structure du programme et de son utilisation. On décrira les classes et fonctions majeures du code, en expliquant par exemple le but de la fonction, ses paramètres et ce qu'elle retourne.

Au niveau de la simulation, une petite étude du comportement thermique de l'ailette devra être réalisée. Plusieurs scénarios seront simulés et ils permettront de montrer l'impact sur l'évacuation de chaleur du dispositif en fonction de la longueur L_x ou encore sur l'utilisation ou non du ventilateur. On expliquera aussi comment le programme C++ est lancé (ligne de commande, fichier de config,...) et quelles sont les données qui ont été générées. De plus, des vérifications du code de calcul seront appréciées, comme par exemple :

— dans le cas stationnaire, une comparaison avec la solution analytique
— dans le cas instationnaire, montrer que l'on converge bien vers la solution stationnaire

Ces résultats numériques seront illustrés à l'aide de graphiques et d'images.