

T H E R M O P T I M ®

MANUEL

DE

RÉFÉRENCE

TOME IV

PILOTES EXTERNES

DIMENSIONNEMENT TECHNOLOGIQUE

NON NOMINAL

EDITEUR DE STRUCTURES PRODUCTIVES

VERSIONS 2.61 A 2.81

© R. GICQUEL NOVEMBRE 2012

SOMMAIRE

1 INTRODUCTION.....	4
1.1 DIMENSIONNEMENT TECHNOLOGIQUE ET REGIME NON-NOMINAL.....	4
1.2 STRUCTURES PRODUCTIVES ET BILANS EXERGETIQUES.....	5
1.3 VERSIONS DE THERMOPTIM CORRESPONDANTES	5
2 PILOTAGE DE THERMOPTIM.....	6
2.1 PRESENTATION GENERALE	6
2.2 IMPLEMENTATION INFORMATIQUE	7
2.2.1 <i>Création de la classe, interface visuelle.....</i>	7
2.2.2 <i>Reconnaissance des noms des composants</i>	7
2.2.3 <i>Calculs effectués et affichage.....</i>	8
2.2.3 <i>Chargement d'un pilote</i>	9
3 DIMENSIONNEMENT TECHNOLOGIQUE	10
3.1 DIMENSIONNEMENT TECHNOLOGIQUE DES COMPOSANTS	10
3.1.1 <i>Echangeurs de chaleur.....</i>	10
3.1.1.1 Méthode de calcul	10
3.1.1.2 Echangeurs multizones.....	11
3.1.1.3 Paramétrage géométrique	11
3.1.1.4 Corrélations pour le calcul des coefficients d'échange thermiques	12
3.1.2 <i>Compresseurs.....</i>	12
3.1.2.1 Compresseurs volumétriques	12
3.1.2.2 Turbocompresseurs	13
3.1.3 <i>Turbines</i>	14
3.1.4 <i>Détendeurs</i>	14
3.2 IMPLEMENTATION INFORMATIQUE	15
3.2.1 <i>Principes généraux.....</i>	15
3.2.2 <i>Structure retenue.....</i>	15
3.2.2.1 Compresseurs volumétriques	16
3.2.2.2 Echangeurs de chaleur.....	17
3.2.3 <i>Ecran d'affichage technologique et de simulation</i>	21
3.2.4 <i>Pilote générique pour la création des écrans technologiques.....</i>	21
3.3 EXEMPLE DE DIMENSIONNEMENT D'ECHANGEUR	22
3.3.1 <i>Présentation et résultats obtenus</i>	22
3.3.2 <i>Conception du pilote</i>	24
3.3.2.1 Initialisations	24
3.3.2.2 Calculs.....	25
3.3.2.3 Sauvegardes	25
4 NON-NOMINAL.....	26
4.1 EXEMPLE.....	26
4.1.1 <i>Résultats obtenus.....</i>	27
4.1.2 <i>Conception du pilote</i>	27
4.1.2.1 Initialisations	27
4.1.2.2 Calculs.....	29
4.1.2.3 Sauvegardes	30
4.2 UTILISATION DU MODELE POUR SIMULER LE REMPLISSAGE D'UN STOCKAGE D'AIR COMPRI ME.....	30
4.3 UTILISATION DU SOLVEUR MINPACK POUR RESOUDRE DES SYSTEMES D'EQUATIONS NON-LINEAIRES	31
4.3.1 <i>Présentation de minPack.....</i>	31
4.3.2 <i>Mise en œuvre de minPack.....</i>	32
4.3.3 <i>Exemple.....</i>	32
5 METHODOLOGIE D'IDENTIFICATION DES PARAMETRES DE COMPOSANTS	34
5.1 IMPORTATION DE DONNEES EXPERIMENTALES	34
5.1.1 <i>Cas particulier d'un évaporateur</i>	35

5.1.2 Cas particulier d'un compresseur <i>VOLUMETRIQUE</i>	36
5.2 AUTOMATISATION DES SIMULATIONS DE SERIES DE FICHIERS DE PROJET	36
5.2.1 Modification des fichiers de projet.....	37
5.2.2 Simulation d'une série de fichiers de projet	37
5.3 Analyse de l'influence des paramètres des corrélations des échangeurs	38
5.3.1 Méthodologie de l'identification.....	38
5.3.2 Automatisation des simulations.....	38
5.3.3 Exploitation du fichier simul.txt.....	40
5.3.4 Méthode d'identification.....	40
5.5 REFERENCES	40
6 EDITEUR DE STRUCTURES PRODUCTIVES	41
6.1 UNITES PRODUCTIVES OU DISSIPATIVES	41
6.2 JONCTIONS ET EMBRANCHEMENTS	42
6.3 AUTOMATISATION DE LA CREATION DE LA STRUCTURE PRODUCTIVE	42
6.4 BILANS EXERGETIQUES DE COMPOSANTS	43
6.5 GENERATION DES BILANS EXERGETIQUES	44
ANNEXE 1 : CLASSES EXTPILOT, POINTTHOPT, CORPSTHOPT	45
CLASSE EXTPILOT	45
CLASSE POINTTHOPT	47
CLASSE CORPSTHOPT	47
CLASSES DE DIMENSIONNEMENT TECHNOLOGIQUE	49
CLASSES REPRESENTATIVES DES TURBINES	49
<i>TechnoSimpleTurb</i>	49
<i>TechnoMultiStageTurb</i>	50
<i>TechnoTurb</i>	50
<i>Classe MultiStageMappedTurbine</i>	51
CLASSE TECHNO NOZZLE	52
CLASSE TECHNO TURBOCOMPR	52
CLASSE MULTISTAGEMAPPEDTURBOCOMPR	53
<i>Remarques sur les valeurs de référence des fichiers de cartographie</i>	54
ANNEXE 2 REPRESENTATION DES COMPOSANTS SOUS FORME D'UNITES PRODUCTIVES ..	55
COMPOSANTS EXTENSEURS D'EXERGIE	55
COMPOSANTS REDUCTEURS D'EXERGIE.....	56

© R. GICQUEL 1997 - 2012. Toute représentation ou reproduction intégrale ou partielle faite sans autorisation est illicite, et constitue une contrefaçon sanctionnée par le Code de la propriété intellectuelle. Avertissement : les informations contenues dans ce document peuvent faire l'objet de modifications sans préavis, et n'ont en aucune manière un caractère contractuel.

1 INTRODUCTION

Dans ce tome 4, nous présentons une série de développements qui ont été réalisés d'une part pour permettre de compléter le noyau de Thermoptim afin de pouvoir aborder des **modélisations beaucoup plus complexes** que celles qui ont été présentées dans le reste du manuel de référence, et d'autre part de générer automatiquement des **structures productives** afin notamment de faciliter l'établissement de bilans exergétiques.

1.1 Dimensionnement technologique et régime non-nominal

Les développements en matière de modélisation permettent de :

- piloter les calculs effectués par Thermoptim en prenant la main sur le moteur de recalcul automatique
- réaliser des dimensionnements technologiques et effectuer des études de comportement d'un système en régime non-nominal.

Compte tenu de l'étendue du domaine correspondant à ces problématiques, nous ne pouvons en aucun cas prétendre leur apporter des réponses exhaustives : nous limitons notre ambition à présenter, autour de quelques exemples relativement simples, le potentiel offert par les pilotes et les outils de dimensionnement technologique et de non-nominal.

Nous ne pouvons, à ce stade de nos travaux, que montrer la voie, laissant aux utilisateurs confirmés qui auront à résoudre des problèmes précis, le soin de bâtir par eux-mêmes les modèles pertinents. Les classes que nous proposons ne doivent être considérées que comme des ébauches destinées à être affinées par les utilisateurs souhaitant étudier le dimensionnement technologique et le comportement des systèmes en régime non-nominal, et non pas comme des modèles parfaitement validés. Notre objectif est simplement de faciliter leur démarche en leur fournissant de quoi démarrer. Pour introduire de nouvelles classes, il suffit soit de sous-classer celles qui existent, soit d'en créer sur le même modèle et de les placer dans extThopt.zip ou extUser.zip pour la version 1.7 et extThopt2.zip ou extUser2.zip pour la version 2.7.

Soulignons qu'il s'agit d'un sujet extrêmement intéressant quoique particulièrement difficile, mal connu et peu traité jusqu'ici, sans doute faute de disposer d'un outil approprié, mais essentiel si l'on veut vraiment comprendre comment se comportent les machines réelles. Ce document étant un manuel de référence du progiciel, nous n'avons pas pu y développer beaucoup les analyses thermodynamiques. Nous renvoyons au tome 3 du livre Systèmes Energétiques le lecteur intéressé par plus développements à ce sujet.

Jusqu'en 2008, la finesse des modèles de composants utilisés dans Thermoptim était limitée : les modèles phénoménologiques construits permettaient certes d'étudier le cycle thermodynamique de la technologie étudiée, mais pas d'en effectuer un dimensionnement technologique précis, ni d'en simuler les performances en régime de fonctionnement non-nominal, ces deux dernières problématiques étant beaucoup plus complexes que la première. Pour pouvoir traiter ces questions, nous avons été conduits à **distinguer deux niveaux dans les modèles** (leur dénomination a été choisie pour être la plus claire possible, mais elle n'est pas usuelle) :

- les **modèles phénoménologiques**, les seuls jusqu'ici mis en œuvre dans Thermoptim, donnent accès au calcul des cycles thermodynamiques, indépendamment du choix d'une technologie de composants particulière ;
- les **modèles de dimensionnement technologique et de simulation en régime non-nominal** fournissent non seulement les mêmes résultats que les précédents, mais, grâce à la prise en compte d'équations complémentaires spécifiques aux technologies choisies, permettent en plus de dimensionner précisément les divers composants et, une fois le dimensionnement technologique réalisé, d'étudier le comportement du système considéré en dehors des conditions de fonctionnement pour lesquelles il a été dimensionné.

De tels modèles sont par exemple nécessaires lorsque l'on veut évaluer les performances d'une installation existante, qui fonctionne souvent dans des conditions différentes de celles pour lesquelles elle a été dimensionnée. Cette problématique d'audit (en vue en particulier de proposer des améliorations) intéresse un nombre croissant d'organismes, industriels ou autres.

L'analyse des systèmes énergétiques en régime non-nominal pose de nombreux problèmes beaucoup plus difficiles à résoudre que ceux que l'on rencontre lorsque l'on se contente d'étudier les cycles thermodynamiques sous l'angle purement phénoménologique au point nominal, indépendamment des choix technologiques.

Précisons bien que ce que nous appelons régime non-nominal correspond au fonctionnement stabilisé d'une installation pour des conditions opératoires différentes de celles pour lesquelles elle a été dimensionnée : il ne s'agit pas d'étudier le régime transitoire rapide résultant par exemple des actionneurs d'une régulation.

Le dimensionnement technologique des composants nécessite d'affiner les modèles phénoménologiques utilisés dans le noyau de Thermoptim, en les complétant de manière appropriée.

Comme nous le verrons dans ce tome, les implémentations de ces nouvelles fonctionnalités de Thermoptim sont pour l'essentiel effectuées sous forme de classes externes, ce qui offre une grande latitude aux utilisateurs pour les personnaliser (cf. tome 3 du manuel de référence).

La version classique phénoménologique de Thermoptim reste inchangée : elle est seulement complétée par une couche logicielle permettant de prendre en compte les modèles de dimensionnement technologique et de non-nominal.

Les classes de dimensionnement technologique complètent celles du noyau en permettant la prise en compte des équations propres aux choix technologiques effectués, comme par exemple le calcul du coefficient d'échange thermique U d'un échangeur de chaleur, ou bien le débit et le rendement isentropique d'un compresseur. Le pilote assure quant à lui la coordination systémique des calculs relatifs aux différents composants qui entrent en jeu. Il recherche un ensemble de variables de couplage cohérent avec le problème physique et technologique posé, et permet ainsi de modifier le paramétrage du projet Thermoptim.

Précisons que toutes les options de paramétrage offertes dans les écrans habituels (phénoménologiques) de Thermoptim ne sont pas nécessairement valides en régime non-nominal, et qu'il importe donc de prévoir, dans les classes externes que l'on développe, des tests destinés à vérifier leur conformité aux hypothèses des modèles retenus.

1.2 Structures productives et bilans exergétiques

Nous avons commencé à implémenter dans Thermoptim un nouvel environnement permettant d'éditer des structures productives au sens de Valero¹. Une structure productive est un graphe permettant de représenter les produits ou consommations d'exergie des différentes unités d'un système énergétique. Selon Valero, ce graphe, qui décrit la distribution des ressources exergétiques, présente un grand intérêt car il constitue une étape essentielle pour pouvoir générer le modèle thermo économique du système considéré. Il permet aussi d'en dresser des bilans exergétiques très précis.

Nous nous sommes contentés d'implémenter dans Thermoptim un nouvel éditeur permettant de concevoir des structures et de les relier au simulateur et à l'éditeur de schémas pour pouvoir calculer proprement les allocations exergétiques de chaque composant et représenter les flux d'exergie échangés. Même si cette tâche est envisagée pour la suite, rien n'a encore été fait pour générer le modèle thermo économique à partir de la structure productive.

1.3 Versions de Thermoptim correspondantes

Ces nouvelles fonctionnalités ont été implémentées dans les versions de Thermoptim suivantes :

- structures productives : 1.6 et 2.6
- dimensionnement technologique et non-nominal : 1.7 et 2.7
- structures productives, dimensionnement technologique et non-nominal : 2.8

¹ VALERO, A., SERRA, L., UCHE, J., Fundamentals of thermoeconomics, lectures 1-3, Euro summer course on sustainable assesment of clean air technologies, téléchargeable sur le site du CIRCE

2 PILOTAGE DE THERMOPTIM

2.1 Présentation générale

Piloter Thermoptim a deux principales applications :

- d'une part faciliter la mise au point des classes externes en les testant au fur et à mesure qu'elles sont définies
- d'autre part donner accès à l'ensemble des bibliothèques non protégées pour différents usages (coordonner les calculs de différents composants couplés entre eux, effectuer des traitements externes entre deux recalculs, guider un utilisateur dans un module de formation...).

Pour rendre possible le pilotage externe, la classe principale du simulateur, *Projet*, peut être sous-classée, et un certain nombre de méthodes non protégées peuvent être surchargées pour définir des traitements complémentaires de ceux du noyau. Le détail de ces méthodes est donné en annexe du tome 3.

Il y a deux moyens de piloter Thermoptim : soit totalement à partir d'une application externe qui instancie le progiciel et le paramètre, soit partiellement, pour un projet donné. Un exemple relevant du premier cas est donné dans le tome 3, où il sert à créer les classes externes et les tester depuis l'environnement de développement des classes externes présenté à la section 6, qui explique comment utiliser un logiciel en freeware pour disposer d'un environnement de travail convivial. Nous n'en parlerons pas particulièrement dans ce tome.

Dans le second cas, auquel nous nous intéresserons ici, on associe une classe de pilotage spécifique à un projet donné, et cette classe est instanciée lors du chargement du projet. Elle permet alors de coordonner les recalculs de ce projet selon des règles particulières. La classe de pilotage est une extension de *extThopt.ExtPilot*, qui dérive de *rg.thopt.PilotFrame*. Pour pouvoir l'associer au projet, il faut commencer par en faire une classe externe pour qu'elle soit chargée dans Thermoptim lors de son lancement. Une fois le projet ouvert, il faut sélectionner la classe de pilotage qui lui est associée par la ligne "Ecran de pilotage" du menu Spécial (cf. section 2.2.3). Si le projet est sauvé, le nom de la classe de pilotage est sauvegardé pour qu'elle puisse être instanciée lors d'un chargement ultérieur. Il est possible de sauvegarder les données du pilote au même titre que celles d'un composant externe, ce qui permet d'enregistrer des résultats de simulation qui ne peuvent l'être dans les éléments habituels de Thermoptim.

Comme nous le verrons à l'occasion des exemples qui sont donnés dans ce tome, cette fonctionnalité de Thermoptim est extrêmement puissante, car elle permet de personnaliser l'utilisation de la plupart des éléments du progiciel, ceci au prix d'un travail de programmation tout à fait raisonnable.

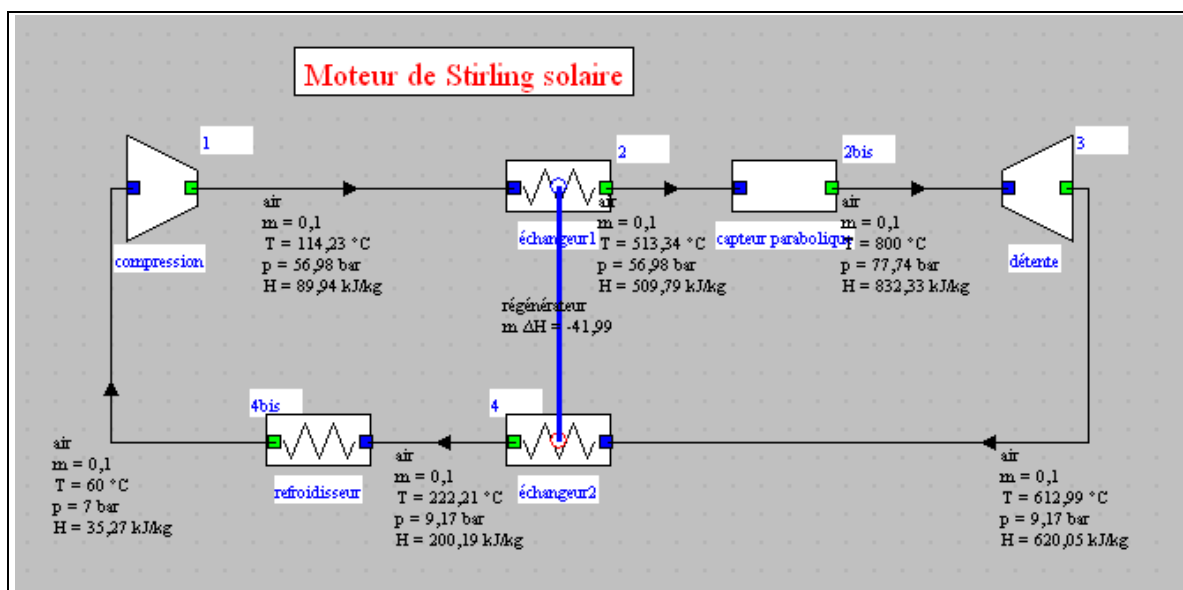


Figure 2.1 : Moteur Stirling solaire

Dans cette section, nous présentons un pilote qui sert à calculer le bilan énergétique d'un moteur Stirling solaire. Il s'agit d'un exemple très simple qui permet d'introduire ce type de classe externe sans qu'il soit nécessaire de se plonger dans le détail d'un modèle thermodynamique complexe.

Un moteur Stirling est un moteur d'un type particulier, qui travaille en système fermé, et met en œuvre une compression refroidie et une détente réchauffée, de telle sorte que les indicateurs de performance usuels de ThermoOptim ne peuvent pas être directement utilisés : l'énergie payante est la somme de la chaleur fournie à la source chaude (dans cet exemple un concentrateur solaire) et de celle apportée pendant la détente.

L'objectif que nous assignons au pilote est de dresser un tableau récapitulatif des énergies mises en jeu dans le moteur, sous forme de chaleur ou de puissance mécanique, et de calculer le rendement du cycle. La figure 2.2 montre le type d'écran que l'on peut imaginer.

	power	heat	
compression	22.325	-18.410	Calculate
expansion	-49.141	33.280	
solar collector		32.254	
external cooling		-20.306	
net power	-26.817		efficiency
total heat supply		65.534	0.410

Figure 2.2 : Ecran du pilote

2.2 Implémentation informatique

2.2.1 Création de la classe, interface visuelle

Pour créer un pilote externe, il suffit de sous-classer `extThopt`. `ExtPilot`.

La réalisation de l'interface visuelle ne pose pas de problème particulier et nous ne la commenterons pas ici. Le constructeur doit se terminer par un `String` précisant le code qui désignera le pilote dans la liste de ceux qui sont disponibles :

```
type="stirling";
```

Il est par ailleurs recommandé de documenter la classe :

```
public String getClassDescription(){
    return "pilot for a simple Stirling motor\n\author : R. Gicquel february 2008";
}
```

2.2.2 Reconnaissance des noms des composants

Il est possible de reconnaître automatiquement les noms des divers composants constituant le modèle, en les triant par type, ce qui confère au pilote une plus grande genericité que si ces noms sont entrés sous forme de `String` dans le code. C'est l'objet des méthodes `init()` et `setupProject()`, qui utilisent les méthodes permettant d'accéder aux noms des composants de l'éditeur de schémas et à la liste des classes externes disponibles (pour la transfo externe représentant le capteur à concentration). Attention toutefois : si un schéma approprié n'est pas ouvert dans l'éditeur, les listes de composants sont vides et les initialisations ne peuvent se faire correctement.

```

public void init(){
    isInitialized=true;
    proj=getProjet();
    setupProject();
    //On récupère la liste et le type des composants présents dans l'éditeur de schémas
    //Retrieves the list and the type of components in the diagram editor
    String[] listComp=proj.getEditorComponentList();
    composant=new String[listComp.length];
    nomComposant=new String[listComp.length];

    //on en extrait les noms des transfos de compression et de détente
    //gets the names of compression et expansion processes
    for(int i=0;i<listComp.length;i++){
        composant[i]=Util.extr_value(listComp[i], "type");
        nomComposant[i]=Util.extr_value(listComp[i], "name");
        if(composant[i].equals("Compression"))compressorName=nomComposant[i];
        if(composant[i].equals("Expansion"))expansionName=nomComposant[i];
    }
    //test de cohérence (des messages d'erreur plus précis seraient souhaitables)
    //consistency test (error messages should be an improvement)
    if(!expansionName.equals("")) && !compressorName.equals("")) && isBuilt)isBuilt=true;
    if(isBuilt)show();//on n'ouvre le pilote que si sa structure est correcte
        //the driver cannot be open if its structure is wrong

void setupProject(){
    //on récupère ici l'instance de la transfo externe du capteur solaire
    //Retrieves the instances of the solar collector external class
    Vector vExt=proj.getExternalClassInstances();//Vector contenant les classes externes
    int j=0;
    for(int i=0;i<vExt.size();i++){
        Object[] obj=new Object[6];
        obj=(Object[])vExt.elementAt(i);
        ExtProcess ep=(ExtProcess)obj[1];
        if(ep instanceof SolarConcentratorCC){
            collector=(SolarConcentratorCC)ep;
            collectorName=collector.getName();
            j++;
        }
    }
    if(j==1)isBuilt=true;//test de cohérence du pilote par rapport au modèle
    //consistency test (error messages should be an improvement)
}

```

2.2.3 Calculs effectués et affichage

Une fois les noms des divers composants identifiés, on accède à leurs propriétés grâce à la méthode `getProperties()` de `Projet`, ce qui permet d'obtenir toutes les valeurs dont on a besoin.

Comme le montre le code correspondant, la réalisation d'un pilote externe ne pose pas de problème particulier. Celui-ci est particulièrement simple et se contente d'effectuer les bilans pour lesquels le mode de calcul par défaut de `Thermoptim` n'est pas approprié. Il serait tout à fait possible de le compliquer légèrement, pour qu'il mette à jour le simulateur avant d'effectuer des recalculs du modèle et de dresser le bilan recherché. Nous présenterons plus loin des exemples mettant en jeu ces fonctionnalités.


```

void bCalculate_actionPerformed(java.awt.event.ActionEvent event){
    if(!isInitialized) init();//la première fois, on initialise, car il faut un constructeur sans argument
                                //pour instancier la classe par le RMI
                                //the initialization is made during the first call, as the constructor
                                //must have no argument because the class is instanciated by the RMI

    String[] args=new String[2];
    args[0]="process";
    args[1]=compressorName;//compression
    Vector vProp=proj.getProperties(args);
    String amont=(String)vProp.elementAt(1);//point amont //inlet point
    String aval=(String)vProp.elementAt(2);//point aval //outlet point
    Double f=(Double)vProp.elementAt(4);
    double deltaUcompr=f.doubleValue();//puissance compresseur //compression power
    f=(Double)vProp.elementAt(12);
    double Qcompr=f.doubleValue();//chaleur compresseur //compressor heat

    args[1]=expansionName;//détente //expansion
    vProp=proj.getProperties(args);
    amont=(String)vProp.elementAt(1);//point amont //inlet point
    aval=(String)vProp.elementAt(2);//point aval //outlet point
    f=(Double)vProp.elementAt(4);
    double deltaUexpan=f.doubleValue();//puissance détente //expansion power
    f=(Double)vProp.elementAt(12);
    double Qexpan=f.doubleValue();//chaleur détente //expansion heat

    args[1]=collectorName;//capteur solaire //solar collector
    vProp=proj.getProperties(args);
    f=(Double)vProp.elementAt(4);
    double solarHeat=f.doubleValue();//chaleur solaire //solar heat

    //calcul des performances globales du moteur et affichages
    //calculates the overall motor energies and displays them on the driver screen
    tauExpan_value.setText(Util.aff_d(deltaUexpan, 3));
    netPower_value.setText(Util.aff_d(deltaUexpan+deltaUcompr, 3));
    tauCompr_value.setText(Util.aff_d(deltaUcompr, 3));
    Q_value.setText(Util.aff_d(Qexpan+solarHeat, 3));
    eta_value.setText(Util.aff_d((-deltaUexpan-deltaUcompr)/(Qexpan+solarHeat), 3));
    expanHeat_value.setText(Util.aff_d(Qexpan, 3));
    comprHeat_value.setText(Util.aff_d(Qcompr, 3));
    solarHeat_value.setText(Util.aff_d(solarHeat, 3));
    extCooling_value.setText(Util.aff_d(-(Qexpan+solarHeat+deltaUexpan+deltaUcompr+Qcompr), 3));

```

2.2.3 Chargement d'un pilote

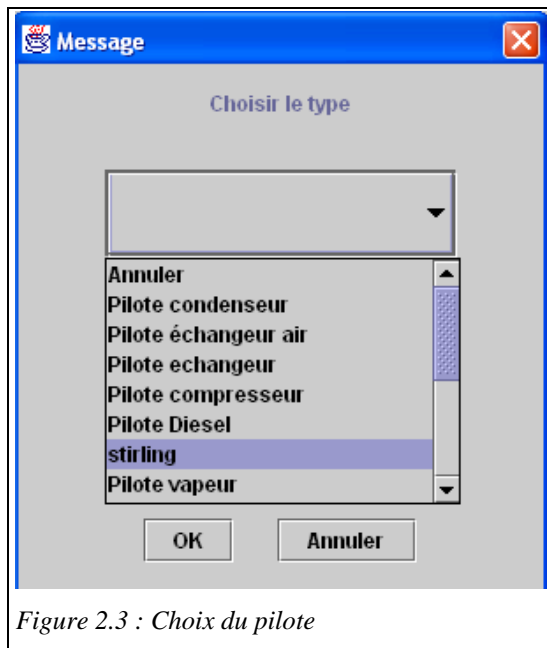


Figure 2.3 : Choix du pilote

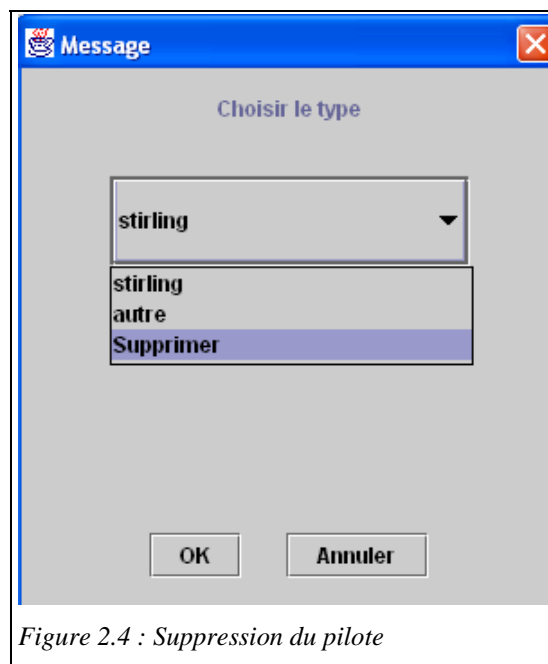


Figure 2.4 : Suppression du pilote

Le chargement d'un pilote se fait en activant la ligne "Ecran de pilotage " du menu "Spécial" du simulateur. Un menu déroulant propose alors la liste des pilotes disponibles (figure 2.3). Sélectionnez celui que vous désirez charger (ici "stirling"), puis validez.

Lorsqu'un pilote est déjà chargé, la ligne "Ecran de pilotage" vous propose l'écran de la figure 2.4, qui vous permet de revenir au pilote chargé (ligne avec son nom), d'en sélectionner un autre, ou de supprimer l'existant sans le remplacer. Un seul pilote peut être associé à un projet.

3 DIMENSIONNEMENT TECHNOLOGIQUE

Pour pouvoir effectuer des dimensionnements technologiques, nous avons introduit de nouveaux écrans, complémentaires de ceux qui permettent d'effectuer la modélisation phénoménologique. Ils peuvent être activés à partir du bouton "tech. design" placé dans les écrans habituels des différents composants.

Ces écrans permettent de définir les caractéristiques géométriques représentatives des différentes technologies utilisées, ainsi que les paramètres nécessaires pour le calcul de leurs performances. Pour un composant donné, ils dépendent bien évidemment du type de technologie retenu.

Dans la section suivante, nous présentons de manière synthétique les principes de modélisation retenus pour les principaux échangeurs, pour les compresseurs, pour les turbines et pour les détendeurs.

3.1 Dimensionnement technologique des composants

3.1.1 Echangeurs de chaleur

Les modifications les plus importantes concernent les échangeurs de chaleur, les versions antérieures de ThermoOptim ne déterminant que le produit UA du coefficient global d'échange thermique U par la surface A de l'échangeur, sans que les deux termes soient évalués séparément. Pour pouvoir dimensionner un échangeur, c'est-à-dire calculer sa surface, il faut d'une part choisir sa configuration géométrique, et d'autre part calculer U , qui dépend de cette configuration, des propriétés thermophysiques des fluides, et des conditions opératoires.

Rappelons que l'approche que nous avons retenue dans ThermoOptim n'est pas conventionnelle, mais qu'elle est cohérente avec d'autres approches utilisées en thermique des échangeurs, et se révèle tout à fait féconde pour l'étude de systèmes complexes : un échangeur assure le couplage entre deux transfos "échange" représentant l'une le fluide chaud qui se refroidit, et l'autre le fluide froid qui se réchauffe. Il en résulte que la définition des configurations d'écoulement et de la géométrie de l'échangeur se fait au niveau de chaque transfo, et non pas de manière globale.

3.1.1.1 Méthode de calcul

La méthode de calcul des échangeurs en régime non nominal retenue est basée sur celle du NUT. Lorsque l'on connaît ses deux températures d'entrée et ses débits, un échangeur peut être représenté par un quadripôle, dont la formulation matricielle généralisée peut s'exprimer de diverses manières, selon les températures connues (chapitre 5 du tome 1 du livre Systèmes énergétiques).

En dimensionnement, le calcul d'un échangeur se fait en trois étapes :

- connaissant les températures d'entrée et de sortie et les débits des deux fluides, on commence par déterminer l'efficacité ε , et on en déduit UA par la méthode du NUT
- on estime alors U par des corrélations dépendant de la configuration d'écoulement et de la géométrie de l'échangeur
- le calcul de la surface A s'en déduit immédiatement : $A = UA/U$

En régime non-nominal, si l'on connaît les températures d'entrée et les débits des deux fluides, la surface A de l'échangeur et sa géométrie (configurations d'écoulement et paramètres technologiques), le calcul se fait en trois étapes :

- détermination de U par des corrélations dépendant de la configuration d'écoulement et de la géométrie de l'échangeur
- calcul de UA , produit de U et de A , puis de NUT
- détermination de l'efficacité de l'échangeur par la méthode du NUT, et calcul des températures de sortie

Le cas le plus naturel est celui où l'on connaît les températures d'entrée des deux fluides, T_{ce} et T_{fe} , et leurs débits m_c et m_f , et où l'on cherche à savoir quelles seront les températures de sortie des deux fluides T_{cs} et T_{fs} lorsque les variables connues changent. Si l'on est capable de déterminer la valeur de U , il est possible de calculer R , puis NUT , d'en déduire ϵ , et de déterminer T_{cs} et T_{fs} .

Mais d'autres cas peuvent aussi être résolus : connaissant T_{fe} , T_{fs} , m_c , m_f et U , on peut calculer R , puis NUT , d'en déduire ϵ , et de déterminer les températures T_{ce} et T_{cs} .

Le premier cas pouvait d'ailleurs être déjà calculé par la version phénoménologique de ThermoOptim : le mode de calcul "non nominal" de l'écran de l'échangeur permet en effet de calculer, par la méthode du NUT , l'échangeur en régime non nominal si au moins deux températures sont imposées. Comme indiqué dans la section du tome 2 du manuel de référence consacrée aux échangeurs, ce mode de calcul s'applique à l'étude d'un échangeur déjà défini, dont on cherche à comprendre comment il se comporte en dehors des conditions retenues pour son dimensionnement. ThermoOptim effectue une mise à jour des liens amont de l'échangeur à partir des transfos, puis effectue le calcul des températures aval et équilibre le bilan sur le plan enthalpique. Les points et les transfos associés au module sont mis à jour en fonction des résultats.

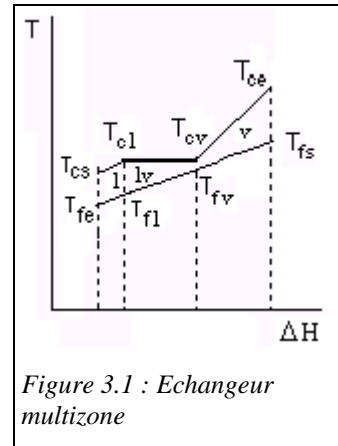


Figure 3.1 : Echangeur multizone

Il est donc possible d'utiliser ce mode de calcul en modifiant la valeur de UA comme l'illustrera l'exemple traité section 4.

3.1.1.2 Echangeurs multizones

Le calcul des échangeurs multizones est plus complexe que celui des échangeurs simples, mais il repose sur le même principe. Considérons en effet le cas d'un condenseur de machine frigorifique dont le diagramme température / enthalpie est représenté figure 3.1.

Le fluide frigorigène sortant du compresseur commence par être désurchauffé ($T_{ce}-T_{cv}$), puis il est condensé à température constante, et enfin légèrement sous-refroidi ($T_{cl}-T_{cs}$). Dans ce cas, le calcul doit être mené pour chacune des zones, la surface de l'échangeur étant la somme des surfaces des zones. Bien évidemment, les coefficients d'échange thermique sont très différents dans ces trois zones.

En régime non-nominal, la surface totale reste constante, mais sa répartition entre les trois zones varie. Un calcul précis demande donc de chercher la solution qui correspond à la même surface totale et qui respecte toutes les autres contraintes thermiques et thermodynamiques.

3.1.1.3 Paramétrage géométrique

Le paramétrage géométrique des échangeurs est un sujet difficile. Nous nous contentons ici d'introduire les grandeurs utilisées, sans discuter des raisons de leur choix.

Figure 3.2 : Condensor technological design screen. L'interface est intitulée "refroidissement". Elle contient des champs de saisie pour les paramètres géométriques : "free flow area" (0.0027), "hydr. diameter" (0.0013), "length" (0.06), "surface factor" (4) et "fin efficiency" (0.8). À droite, il y a un menu déroulant "ext_tube | Colburn correlation for single phase flow outside tubes" et un bouton "correlation settings". En bas à droite, des champs indiquent les résultats : "local ΔP loss coeff." (0), "pressure drop" (0.000347) et "friction factor" (0.332839).

Figure 3.2 : Condensor technological design screen

L'étude du calcul des coefficients d'échange thermique et des pertes de charge montre que, outre la surface de l'échangeur A , deux grandeurs géométriques jouent un rôle particulièrement important : la section de passage dévolue au fluide A_c , et le diamètre hydraulique d_h . Lorsque les coefficients d'échange thermique des fluides sont très différents, on a recours à des dispositifs divers comme des ailettes pour compenser l'écart entre leurs

valeurs. On parle alors de surfaces étendues, qui peuvent être caractérisées par un facteur de surface f et un rendement d'ailette η_0 .

Ces quatre paramètres sont ceux qui ont été retenus dans Thermoptim pour caractériser les échanges thermiques au niveau de chaque fluide. On leur ajoute la longueur de l'échangeur pour certains calculs comme les pertes de charge.

Dans l'écran de dimensionnement technologique des échangeurs, les conventions suivantes sont adoptées :

- le type de configuration (ici "cond") est sélectionné dans une liste déroulante
- "free flow area" représente la section de passage A_c
- "hydr. diameter" est le diamètre hydraulique habituel d_h
- "length" est la longueur de l'échangeur (utilisée pour le calcul du nombre d'ébullition Bo et pour le calcul des pertes de charge, non encore implémenté dans le cas diphasique)
- "surface factor" est le facteur de surface f pour les surfaces étendues
- "fin efficiency" est le rendement d'ailettes η_0

3.1.1.4 Corrélations pour le calcul des coefficients d'échange thermiques

TABLEAU 12.1	TYPE	CORRELATION	TYPE DE FLUIDE
"int_tube"	intérieur tubes	Mac Adams	monophasique
"ext_tube"	extérieur tubes	Colburn (Dittus Boettler)	monophasique
"cond"	condensation int. tubes	Shah/Bivens	diphasique
"cond_ext_hor"	cond. ext. tubes horiz.	B1540 T/I	diphasique
"evap"	évaporation	Gungor Winterton	diphasique
"air_coil"	batterie à air	Morisot	air
"cool_tower"	refroidissement évaporatif	Colburn	air humide
"flooded"	noyé	générique	diphasique
"plate"	à plaques	générique	monophasique

Une des principales difficultés est le calcul du coefficient d'échange global U , qui dépend des coefficients d'échange thermiques h de chaque fluide, par la formule générale ci-dessous. De nombreuses corrélations ont été proposées dans la littérature, et le choix de celle qui est le mieux adaptée n'est pas toujours évident.

$$\frac{1}{U_c A_c} = \frac{1}{A_c \eta_{0,c} h_c} + \frac{e}{A \lambda} + \frac{1}{A_f \eta_{0,f} h_f}$$

Les options offertes par défaut par Thermoptim sont données dans le tableau ci-dessus. Il récapitule les configurations proposées, en indiquant leur type, les corrélations utilisées, et le type de fluide auxquelles elles correspondent. Il est bien sûr possible d'en introduire d'autres sous forme de classes externes additionnelles.

Attention à un point important : pour que les calculs aient un sens, il faut impérativement que les unités du projet Thermoptim soient exprimées dans le système SI, c'est-à-dire que le débit soit exprimé en kg/s. Vérifiez bien ce point, en faisant de préférence un test sur l'unité dans le pilote.

3.1.2 Compresseurs

3.1.2.1 Compresseurs volumétriques

Un compresseur volumétrique est défini géométriquement par sa cylindrée, et ses paramètres technologiques permettent de calculer ses rendements volumétrique et isentropique, en fonction de sa vitesse de rotation, de son facteur de charge, et des conditions d'aspiration et de refoulement.

Les modèles implémentés dans Thermoptim sont basés sur l'hypothèse que le comportement des compresseurs volumétriques peut être représenté avec une précision raisonnable par deux paramètres : leur rendement volumétrique λ qui caractérise le volume balayé réel, et leur rendement isentropique classique η_s .

$$\lambda = a_0 - a_1 \frac{P_{ref}}{P_{asp}}$$

$$\eta_s = K_1 + K_2 \cdot \left(\frac{P_{ref}}{P_{asp}} - R_1 \right)^2 + \frac{K_3}{\frac{P_{ref}}{P_{asp}} - R_2}$$

Le calcul d'un compresseur correspond aux étapes ci-dessous :

- les rendements volumétrique et isentropique λ et η_s sont calculés à partir de leurs équations
- si l'on connaît la vitesse de rotation, la cylindrée et le rendement volumétrique, le débit volumique peut être calculé :

$$\dot{V} = \frac{\lambda N V_s}{60}$$

- connaissant la masse volumique à l'aspiration v , on en déduit le débit massique :

$$\dot{m} = \frac{\dot{V}}{v} = \frac{\lambda N V_s}{60 v}$$

En mode dimensionnement, on détermine la vitesse de rotation ou la cylindrée permettant de fournir le débit souhaité. Le calcul est effectué en prenant en compte les pressions d'entrée et de sortie et la valeur du débit saisi dans le champ du débit du compresseur.

En régime non-nominal, le rapport de compression détermine λ et η_s , ce qui fixe le débit et la température de sortie compresseur. La séquence des calculs est la suivante :

- 1) connaissant les conditions d'entrée et de sortie, le rapport de compression P_{ref}/P_{asp} et le volume v à l'aspiration sont mis à jour ;
- 2) les rendements volumétrique et isentropique λ et η_s et le débit réel sont calculés connaissant la vitesse de rotation N ;
- 3) le débit volumique \dot{V} s'en déduit ;
- 4) le compresseur impose $\dot{M} = v \dot{V}$ et le propage aux composants connectés ;
- 5) le rendement isentropique η_s permet alors de calculer le travail utile mis en jeu.

3.1.2.2 Turbocompress eurs

Les turbocompresseurs peuvent être modélisés de plusieurs manières. Nous nous baserons sur une approche par similitude faisant appel à deux caractéristiques : le facteur de débit ϕ , et le facteur d'enthalpie ψ ,

et nous nous limiterons ici à faire l'hypothèse simplificatrice que leurs caractéristiques réduites (ψ, ϕ) et (η, ϕ) peuvent être représentées par des courbes simples du type parabole (figure 3.3).

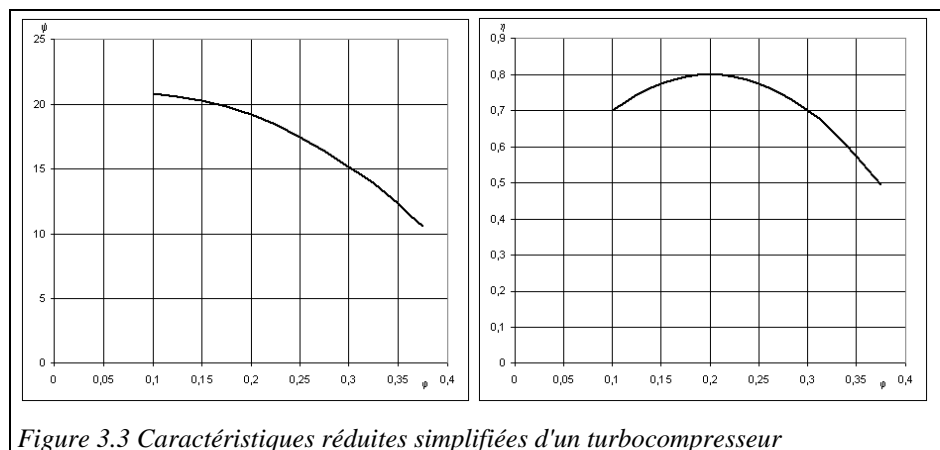


Figure 3.3 Caractéristiques réduites simplifiées d'un turbocompresseur

$$\varphi = \frac{(M_a)_c}{(M_a)_u} = \frac{240 \text{ m r } T_a}{\pi^2 D^3 P_a N}$$

$$\Psi = \frac{|\Delta h_s|}{1/2 U^2} = \frac{7 \text{ 200 } |\Delta h_s|}{\pi^2 D^2 N^2}$$

D'autres modèles sont fournis en annexe.

3.1.3 Turbines

Une turbine est définie géométriquement par une section, et ses paramètres technologiques permettent de calculer son rendement isentropique et la constante du cône, en fonction des conditions d'aspiration et de refoulement.

Pour le rendement isentropique, il est souvent possible de retenir une équation analogue à la loi des compresseurs volumétriques.

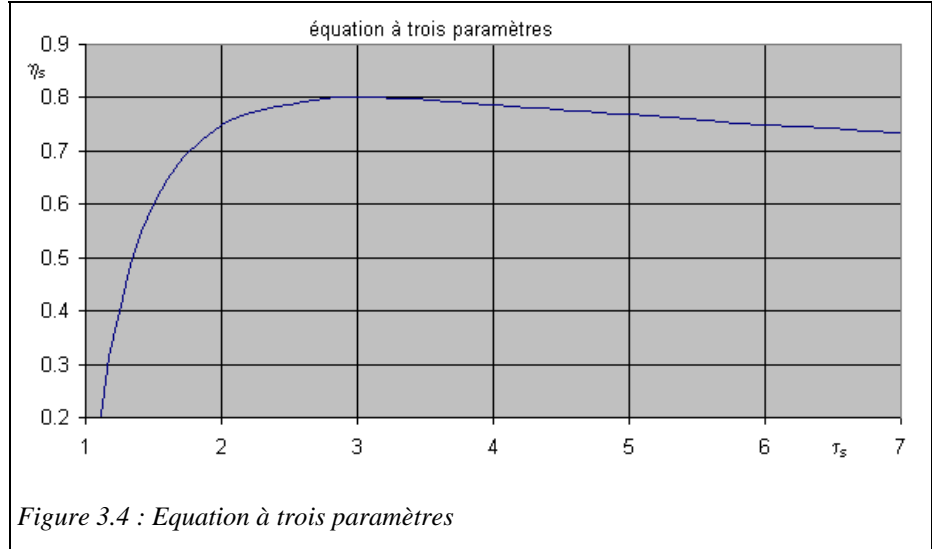


Figure 3.4 : Equation à trois paramètres

$$\eta_p = \eta_{lim} + (\eta_{max} - \eta_{lim}) \cdot \left(2 \left[\tau_{max} \frac{P_{ref}}{P_{asp}} \right] - \left[\tau_{max} \frac{P_{ref}}{P_{asp}} \right]^2 \right)$$

On notera qu'elle s'exprime linéairement en fonction du rapport de détente et de son carré. Ses paramètres présentent l'intérêt d'avoir un sens physique.

η_{lim} est la valeur asymptotique du rendement isentropique pour les hauts rapports de détente, et η_{max} la valeur maximale du rendement, obtenue pour un rapport de détente égal à τ_{max} .

Le calcul du débit se fait en utilisant la règle de Stodola, qui peut s'exprimer sous la forme suivante, K_0 étant la constante du cône :

$$\frac{\dot{m} \sqrt{T_{in}}}{P_{in}} = K_0 \sqrt{1 - \left[\frac{P_{out}}{P_{in}} \right]^{(k+1)/k}}$$

Généralement, on prendra $k=1$, ce qui correspond à une forme quadratique. Si le débit est choqué, cette relation

se simplifie sous la forme $\frac{\dot{m} \sqrt{T_{in}}}{P_{in}} = \text{Cste}$.

D'autres modèles sont fournis en annexe.

3.1.4 Détendeurs

Le dimensionnement technologique des détendeurs n'est pris en compte que de manière sommaire, car on considère que leur dynamique est beaucoup plus rapide que celle des autres composants du cycle, et qu'en conséquence ils se comportent comme des organes de régulation.

Le seul paramètre introduit à ce niveau est la valeur de la surchauffe.

3.2 Implémentation informatique

Les structures des classes externes permettant de traiter le dimensionnement technologique et le non nominal pour les divers composants de Thermoptim présentent un certain nombre de similitudes avec celles qui ont été exposées au tome 3, mais aussi quelques différences qui s'expliquent par leurs caractéristiques propres, et que nous présenterons en temps utile.

Les principaux développements ayant porté sur les compresseurs et les échangeurs de chaleur, c'est eux qui seront présentés ici.

3.2.1 Principes généraux

Tout d'abord, il faut garantir la cohérence avec les classes "phénoménologiques" de Thermoptim, qui doivent pouvoir effectuer leurs calculs de la même manière, que les écrans technologiques soient présents ou non.

Ceci implique en particulier que l'affichage de l'écran technologique doit être distinct des écrans "phénoménologiques".

Ensuite, nous avons décidé d'externaliser le plus possible l'implémentation informatique afin que les utilisateurs puissent personnaliser les calculs autant qu'ils le souhaitent. Il est en effet quasiment impossible d'une part de trouver des formulations complètement génériques compte tenu de la multiplicité des solutions technologiques envisageables, et d'autre part de mettre au point des algorithmes suffisamment robustes pour résoudre les jeux d'équations qui peuvent en découler. En externalisant ces classes, les utilisateurs peuvent les adapter relativement aisément aux problèmes spécifiques auxquels ils sont confrontés.

Nous nous sommes donc contentés de mettre à leur disposition une infrastructure de travail leur facilitant significativement la tâche.

Par principe donc, nous considérons que les calculs de dimensionnement technologique et de non nominal doivent être effectués dans des classes externes, de telle sorte que celles que nous avons intégrées dans le noyau servent essentiellement à permettre la communication entre ce dernier et les classes externes.

En effet, les objets du noyau n'étant pas directement accessibles depuis l'extérieur, les échanges d'informations demandent à être faits en suivant des règles un peu contraignantes, que nous avons cherché à rendre aussi souples et conviviales que possibles, davantage encore que ne le permettaient les méthodes de communication présentées dans le tome 3. Les méthodes que nous avons implémentées à cet effet sont documentées en annexe.

3.2.2 Structure retenue

Dans les écrans de composants, le bouton "tech. design" donne accès à l'écran de paramétrage technologique. Les écrans technologiques peuvent être gérés de manière centralisée, à partir de la ligne de menu "Technological design screens" du menu "Technological Design" du simulateur.

Une classe TechnoDesign a été créée comme classe mère des écrans de paramétrage technologique. Elle est sous-classée par les écrans spécifiques de chaque type de composant, selon le principe présenté section 2.3 du tome 3.

Le dimensionnement technologique est effectué dans tous les TechnoDesign par la méthode makeDesign().

Dans les TechnoDesign, des classes dérivées de JPanel permettent de modifier les paramètres en fonction du choix technologique effectué.

Des classes externes mères représentatives des principaux types de composants ont été préparées. Elles sont destinées à être sous-classées en fonction des besoins des utilisateurs.

La seule manière d'instancier les TechnoDesign destinés à être reconnus par le noyau est de le faire dans un pilote externe, puis de les charger dans le Vector vTechno de ce pilote. Dès lors, ils deviennent visibles depuis le noyau et peuvent être associés à leurs composants. Ce choix signifie que **les pilotes externes jouent un rôle prépondérant pour le dimensionnement technologique et le non nominal.**

La sauvegarde des paramètres technologiques est effectuée à la fin des fichiers de projet, à partir du pilote. La dernière ligne doit se terminer par un retour à la ligne ("\n"). Certains TechnoDesign peuvent occuper plusieurs lignes (c'est le cas pour les échangeurs, qui en utilisent une pour les résultats d'ensemble, et une pour chaque fluide, et pour les thermocoupleurs qui en utilisent deux).

Toutes les classes de dimensionnement technologique héritent de la classe extThopt. ExtTechnoDesign qui possède un JPanel appelé JPanel1, dans lequel est conçue l'interface graphique.

Elle dispose de deux constructeurs :

```
public ExtTechnoDesign(Projet proj, String componentName, PointThopt amont, PointThopt aval){
}
```

```
public ExtTechnoDesign(Projet proj, String componentName){
}
```

Le premier suppose que l'utilisateur définisse ses propres PointThopt amont et aval, tandis que le second les instancie automatiquement, à partir des propriétés du projet, ce qui lui donne une plus grande généralité.

Les PointThopt présentés en annexe, qui sont en quelque sorte des clones externes des points du noyau, lui donnent accès aux états amont et aval. Le constructeur instancie un CorpsThopt (voir annexe) pour permettre d'effectuer certains calculs thermodynamiques.

La méthode updateDeltaH() sert à mettre à jour l'enthalpie mise en jeu dans le composant. Cette classe possède aussi une méthode getFlow(), qui permet d'actualiser le débit à partir de celui du noyau.

Nous commentons ici deux exemples d'implémentation, relatives aux compresseurs volumétriques et aux échangeurs, qui seront utilisés dans les exemples de ce tome. Les autres sont construits de manière analogue.

3.2.2.1 Compresseurs volumétriques

La classe VolumCompr comporte :

- une méthode, setupPanel(), qui construit le JPanel pour l'interface graphique de l'écran technologique, qui n'est pas détaillé ici. Cette interface permet d'effectuer les paramètres nécessaires.
- des méthodes génériques qui permettent de calculer le rendement isentropique et le débit, et de les afficher à l'écran.

Les constructeurs sont les suivants :

```
public VolumCompr(Projet proj, String compressorName, PointThopt amont, PointThopt aval){
    super(proj, compressorName, amont, aval);
    this.compressorName=componentName;
    setupPanel();
}
```

```
public VolumCompr(Projet proj, String compressorName){
    super(proj, compressorName);
    this.compressorName=componentName;
    setupPanel();
}
```

Ils font appel aux constructeurs d'ExtTechnoDesign, puis, pour pouvoir utiliser le nom plus spécifique de compressorName, ils l'identifient à componentName, et enfin, ils construisent le JPanel.

Cette classe définit en particulier les deux méthodes de calcul des rendements isentropique et volumique:


```

double getRisentr(){
    K1=Util.lit_d(K1_value.getText());
    K2=Util.lit_d(K2_value.getText());
    K3=Util.lit_d(K3_value.getText());
    R1=Util.lit_d(R1_value.getText());
    R2=Util.lit_d(R2_value.getText());

    double r_compr=aval.P/amont.P;
    double risentr=0.5;
    if(R1*R1+R2*R2==0)risentr=K1+K2/r_compr+K3/r_compr/r_compr;//3 parameter equation
    else risentr=K1+K2*(r_compr-R1)*(r_compr-R1)+K3/(r_compr-R2);
    risentr_value.setText(Util.aff_d(risentr,7));
    return risentr;
}

double getLambdaVol(){
    a0=Util.lit_d(a0_value.getText());
    alpha=Util.lit_d(alpha_value.getText());

    double r_compr=aval.P/amont.P;
    double lambda= a0-alpha*r_compr;
    display_value.setText(Util.aff_d(lambda,7));
    return lambda;
}

```

La sauvegarde des valeurs des paramètres se fait grâce à la méthode `saveCompParameters()`, qui, comme nous l'avons dit plus haut, doit se terminer par un caractère de retour à la ligne.

```

public String saveCompParameters(){
    String h="alpha_value="+alpha_value.getText()+tab
    +"K1_value="+K1_value.getText()+tab
    +"K2_value="+K2_value.getText()+tab
    +"K3_value="+K3_value.getText()+tab
    +"R1_value="+R1_value.getText()+tab
    +"R2_value="+R2_value.getText()+tab
    +"N_value="+N_value.getText()+tab
    +"Vs_value="+Vs_value.getText()+tab
    +"Device_number="+deviceNumber_value.getText()
    +"\\n";//retour à la ligne impératif
    return h;
}

```

3.2.2.2 Echangeurs de chaleur

Une différence notable entre un échangeur et un compresseur est que le second est un composant unique, alors que le premier représente un couplage thermique entre deux transfos échange. En revanche, les équations de couplage thermique sont les mêmes pour tous les types d'échangeurs que Thermoptim peut calculer, avec une exception : les échangeurs multizones se présentent souvent comme des batteries d'échange dont les zones liquide, diphasique et vapeur ont des limites évolutives dans le temps. Il est donc impossible d'affecter à chacune d'entre elles une surface déterminée : il faut les recalculer en permanence, même si la surface totale est connue.

Le TechnoDesign des échangeurs s'appelle TechnoHx. En mode design, il calcule le U et propose donc une valeur de A, surface de l'échangeur. Il effectue ses calculs en estimant l'ensemble des propriétés thermophysiques des fluides. Le chapitre 13 du tome 3 du livre Systèmes Energétiques précise comment celles-ci sont déterminées.

Le TechnoDesign pour les transfos de type Exchange est TechnoExch, qui permettra aussi de calculer les pertes de charge lorsque les méthodes seront implémentées. Pour garantir un traitement homogène entre les échangeurs et les deux transfos qu'il connecte, TechnoHx fait directement appel aux deux TechnoExch associés, qu'il charge sur son écran. Les TechnoExch des transfos d'un échangeur ne sont donc pas instanciées indépendamment de son TechnoHx.

Différentes configurations d'écoulement ont été introduites : elles héritent toutes de FlowConfig et sont sélectionnées au niveau des TechnoExch. Ce sont elles qui assurent le calcul des propriétés thermophysiques et du nombre de Nusselt, notamment pour les échangeurs multizones.

En résumé :

- TechnoHx introduit la surface de l'échangeur, en fait le dimensionnement (makeDesign()), gère les affichages des calculs des propriétés thermophysiques et instancie les TechnoExch froid et chaud
- les TechnoExch froid et chaud font les calculs des propriétés thermophysiques et gèrent les FlowConfig grâce à des ComboBox. Les JPanel des TechnoExch apparaissent dans la fenêtre du TechnoHx.
- les FlowConfig implémentent les corrélations qui permettent de calculer les coefficients d'échange et de pertes de charge
- le SettingsFrame d'un FlowConfig peut permettre de modifier ses paramètres.

Le constructeur de la classe TechnoHx est le suivant. Il diffère du précédent du fait de la structure particulière de la classe. Il commence par récupérer les noms des transfos chaude et froide qu'il apparie, instancie quatre PointThopt pour connaître les états amont et val des deux fluides, et construit leurs écrans technologiques. Il indique ensuite à chacun d'eux quel est son homologue. Enfin, il configure son écran.

```
public TechnoHx(Projet proj, String hXname, PointThopt amontHot, PointThopt avalHot, PointThopt
amontCold, PointThopt avalCold){
    this();
    this.proj=proj;
    this.hXname=hXname;
    String[] args=new String[2];
    args[0]="heatEx";
    args[1]=hXname;
    Vector vProp=proj.getProperties(args);
    hotFluid=(String)vProp.elementAt(0);
    coldFluid=(String)vProp.elementAt(1);

    this.amontHot=amontHot;
    this.amontCold=amontCold;
    this.avalHot=avalHot;
    this.avalCold=avalCold;

    techc=new TechnoExch(proj, this, hotFluid, amontHot, avalHot);
    techf=new TechnoExch(proj, this, coldFluid, amontCold, avalCold);

    techc.setOtherTechnoExch();
    techf.setOtherTechnoExch();

    JLabCompName.setText(hXname);
    hotName.setText(hotFluid);
    coldName.setText(coldFluid);
    setupPanel();
    setupTechHx();
}
```

La classe implémente ensuite toute une série de méthodes permettant d'effectuer les calculs thermodynamiques.

Le constructeur des TechnoDesign des transfos "échange" est analogue à celui des compresseurs. Il en existe en fait trois variantes, leur signature différant du fait de la nécessité pour eux de savoir s'ils sont rattachés à un échangeur (TechnoHx) , à un thermocoupleur (TechnoT) ou ils sont isolés.

```
public TechnoExch(Projet proj, TechnoHx tHx, String exchangeName, PointThopt amont, PointThopt aval){
    this(proj, exchangeName, amont, aval);
    this.tHx=tHx;
}
```

```

public TechnoExch(Projet proj, String exchangeName, PointThopt amont, PointThopt aval){
    super(proj, exchangeName, amont, aval);
    this.exchangeName=componentName;
    setupPanel();
    JLabCompName.setText(exchangeName);
    fc=new IntTubeConfig(this);
    setupListeConf();
}

```

TechnoExch étant la classe de dimensionnement technologique des transfos Echange, elle joue un rôle essentiel dans l'étude des échangeurs.

Les variables globales qu'elle définit sont les suivantes :

```

boolean diph;
TechnoHx tHx;
TechnoExch otherTe;
TechnoTC tTc;
public FlowConfig fc;
SettingsFrame sf;
String[]listeConf;

```

diph permet d'indiquer que le calcul des coefficients d'échange doit être fait en diphasique.

tHx est la classe TechnoHx appelante lorsque le TechnoExch est un élément d'un échangeur.

otherTe est le second TechnoDesign d'un échangeur dans ce cas.

tTc est la classe TechnoTC appelante lorsque le TechnoExch est un élément d'un thermocoupleur.

fc est le FlowConfig définissant la configuration d'écoulement.

sf est l'écran de paramétrage des coefficients des corrélations du FlowConfig

listeConf est la liste des configurations d'écoulement disponibles qui permet à l'utilisateur de spécifier le type de configuration d'écoulement dans lequel circule le fluide, et donc les équations à retenir pour calculer les coefficients d'échange thermiques.

La méthode setupListeConf() construit dynamiquement la liste des FlowConfig lors du chargement de Thermoptim, en fonction des classes externes qui les définissent. Les codes définissant ces configurations d'écoulement et leur descriptif sont chargés dans une liste déroulante pour que l'utilisateur puisse faire son choix. En cas de doute, si la liste est trop étroite, il peut afficher l'écran de paramétrage correspondant en cliquant sur "correlation settings", qui lui permet de lire l'ensemble du descriptif (cf. figure 3.11).

Le constructeur de la classe mère est donné ci-dessous. Il associe au FlowConfig le TechnoExch qui fait appel à lui, et initialise certaines grandeurs nécessaires au calcul du nombre de Nusselt.

```

public FlowConfig (TechnoExch te){
    this.te=te;
    C1=0.023;a_Re=0.8;b_Pr=0.4;c_Visc=0.14;RR=0.001;
    te.JLabel13.setText("length");
    otherTe=te.otherTe;
}

```

Pour rajouter une autre configuration, il suffit de sous-classer FlowConfig et d'attribuer un code qui n'existe pas déjà, puis de placer la classe dans extThopt2.zip ou extUser2.zip. Elle sera automatiquement incluse dans listeConf.

Les pertes de charge sont calculées dans le FlowConfig de la manière suivante.

En régime laminaire, les pertes de charge sont proportionnelles au débit et à la viscosité du fluide : $f = 64/Re$.

Pour des tubes lisses, f est donné par la formule de Blasius ($f = 0,316 Re^{-0,25}$) si $Re < 30\,000$, ou bien par la formule suivante ($0,0032 + 0,221 Re^{-0,237}$) pour des valeurs de Re supérieures.

Pour les tubes rugueux en régime turbulent, ($Re > 2100$) le coefficient de frottement est donné par l'équation de Colebrook :

$$\frac{1}{\sqrt{f}} = -0,868589 \ln \left[\frac{RR}{3,71} + \frac{2,51}{Re \sqrt{f}} \right]$$

$$RR = \frac{\varepsilon}{D}$$

ε = rugosité absolue du tube

On peut montrer que cette équation implicite en f peut être approximée par :

$$\frac{1}{\sqrt{f}} = -0,78173 \ln \left[\left[\frac{RR}{3,71} \right]^{1,1} + \frac{6,9}{Re} \right]$$

Les paramétrages du SettingsFrame permettent d'opter pour un calcul à partir de la formule de Colebrook si cette option est sélectionnée, ou de type Blasius autrement.

En régime diphasique, le calcul des pertes de charge peut se révéler très complexe, notamment du fait qu'il faut commencer par déterminer quelles sont les longueurs des différentes zones de l'échangeur.

Pour la zone d'équilibre liquide-vapeur, nous avons utilisé la formulation approchée pour les GV nucléaires proposée par Herre et Gallori, modifiée pour tenir compte d'un titre d'entrée non nul.

$$\Delta P_{\text{diff}} = \varphi \Delta P_{\text{liq}}$$

$$\varphi = \left[1 + |x_s - x_e| \left(\frac{\rho_l}{\rho_v} - 1 \right) \right] \left[1 + |x_s - x_e| \left(\frac{\mu_l}{\mu_v} - 1 \right) \right]^{-0,2}$$

La perte de charge est ainsi calculée en utilisant ce facteur de renforcement par rapport à l'état saturé liquide.

Afficher les écrans technologiques

nom	type	classe
evaporator	Echangeur de chaleur	TechnoEvaporator
condenser	Echangeur de chaleur	TechnoCondensor
compressor	compression	VolumCompr

Afficher les types observés

nom	type	débit / P (bar)	m Δh / T (°C)
compressor	compression	1,2479	72,08
3	point	11,7158 (bar)	40,4 °C
1	point	1,3422 (bar)	-14,75 °C
air inlet	point	1 (bar)	40 °C

Lire les paramètres

Sauver les paramètres

Calculer le pilote

Stop

types invalidés

types calculables

☒ débit auto

☐ verrouillé

efficacité

énergie utile

énergie payante

Recalculer

Fermer

Figure 3.5 : Ecran de présentation d'ensemble des outils de dimensionnement technologique

3.2.3 Ecran d'affichage technologique et de simulation

Un écran d'affichage technologique et de simulation a été ajouté au simulateur. Il est accessible depuis la ligne "Ecrans de dimensionnement technologique" du menu "Dimensionnement technologique" (figure 3.5) du simulateur, ou en tapant Ctrl T. Il comporte deux tables principales placées dans sa partie gauche. Celle du haut permet d'accéder à l'ensemble des TechnoDesign instanciés par le pilote et chargés dans le Vector vTechno. Un double-clic sur une des lignes ouvre le TechnoDesign sélectionné. Celle en-dessous permet d'avoir un accès rapide à certains points ou transfos de Thermoptim, que l'on qualifie d'observées parce que l'on désire observer leur état thermodynamique.

Les boutons et affichages du bas de l'écran facilitent le recalcul automatique, qui peut être déclenché sans revenir à celui du simulateur.

Les deux boutons "Calculer le pilote" et "Stop" permettent de lancer et d'arrêter les calculs du pilote sans pour autant bloquer l'accès à Thermoptim (on dit qu'il s'exécute dans un thread séparé), ce qui présente l'avantage qu'il est possible d'accéder à toutes les fonctions du progiciel pendant qu'il effectue les opérations programmées dans le pilote. Le bouton "Stop" permet d'interrompre ces calculs en cas de besoin.

Pour pouvoir utiliser cette fonctionnalité, il faut que la méthode de calcul du pilote s'appelle calculatePilot(). Bien évidemment, il est recommandé de ne pas effectuer de modifications du paramétrage pendant que le pilote s'exécute, pour éviter toute interférence.

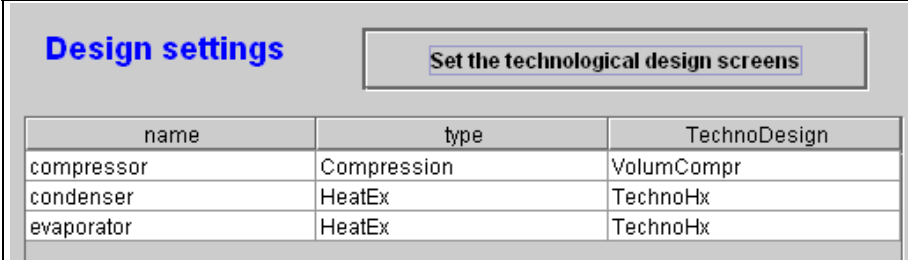
Les deux boutons situés en haut à droite, "Sauver les paramètres" et "Lire les paramètres", vous permettent de sauvegarder un paramétrage technologique réalisé avec un pilote, et de le relire avec un autre semblable, ce qui évite d'avoir à ré-entrer toutes les valeurs chaque fois que l'on change de pilote.

3.2.4 Pilote générique pour la création des écrans technologiques

Nous montrerons dans la section 4 comment créer des écrans technologiques en construisant des pilotes, ce qui suppose un minimum de programmation. Ce travail est impératif lorsqu'on veut effectuer des calculs en régime non-nominal, car il est alors nécessaire de prendre la main sur le moteur de recalcul de Thermoptim.

En revanche, lorsque l'on se contente de vouloir effectuer le dimensionnement technologique d'un projet qui ne met en œuvre que des composants du noyau de Thermoptim, il est possible de créer automatiquement les écrans technologiques en utilisant le pilote générique que nous allons brièvement présenter, ce qui évite d'avoir à en programmer un.

Ouvrez le projet Thermoptim, et chargez le pilote générique en choisissant, dans la liste des pilotes, celui dont l'intitulé est "generic techno design pilot", puis cliquez sur le bouton "Set the technological design screens" (figure 3.6). La liste des composants



name	type	TechnoDesign
compressor	Compression	VolumCompr
condenser	HeatEx	TechnoHx
evaporator	HeatEx	TechnoHx

Figure 3.6 : Ecran du pilote générique (initialisation par défaut)

pour lesquels existent des écrans technologiques apparaît dans la table, avec leur nom, leur type et le nom de la classe d'écran technologique instanciée par défaut : VolumCompr pour le compresseur, et TechnoHx pour les échangeurs de l'exemple de la figure 3.6.

Dans cet exemple, cette initialisation convient pour le compresseur, mais pas pour les échangeurs, le condenseur devant être modélisé par la classe TechnoCondensor, et l'évaporateur par TechnoEvaporator.

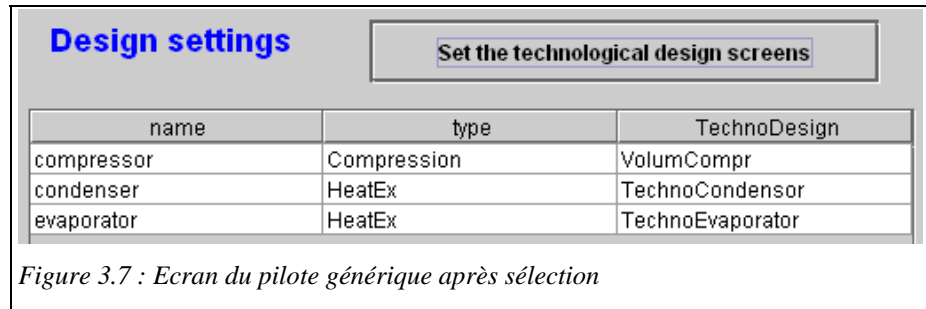


Figure 3.7 : Ecran du pilote générique après sélection

Pour changer la classe du TechnoDesign d'un composant, sélectionnez sa ligne, puis double-cliquez. Un message vous demande de confirmer votre intention, puis vous propose la liste des classes disponibles. Choisissez celle que vous désirez et validez votre choix. Après avoir modifié les classes des deux échangeurs, vous obtenez l'écran de la figure 3.7.

Vous pouvez alors accéder aux écrans technologiques de ces composants soit depuis leurs propres écrans (bouton "tech. Design"), soit à partir de l'écran d'affichage technologique et de simulation présenté section précédente, qui s'ouvre depuis l'écran du simulateur (figure 3.8).

3.3 Exemple de dimensionnement d'échangeur

3.3.1 Présentation et résultats obtenus

Considérons un échangeur à tubes et ailettes destiné à refroidir environ 0,66 kg/mn d'air sortant d'un compresseur à 5 bars et 275 °C grâce à un débit de 1,17 kg/mn d'eau

froide passant dans un serpentin de deux tubes en parallèle. Le diamètre des tubes est de 15 mm, et leur épaisseur de 1,5 mm. L'espacement entre les ailettes est de 3 mm.

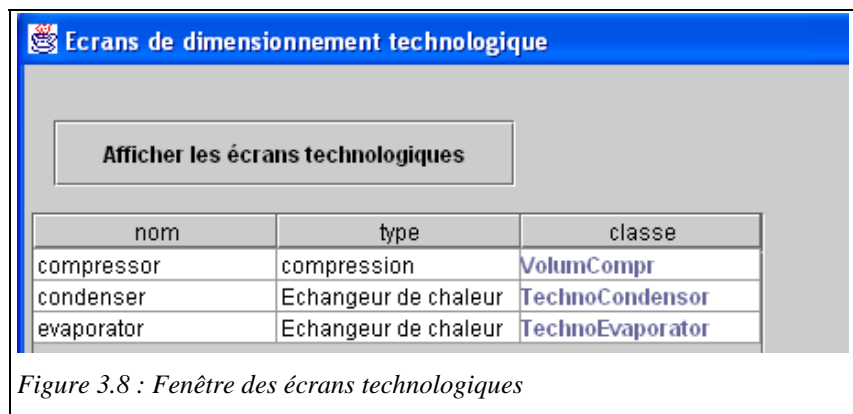


Figure 3.8 : Fenêtre des écrans technologiques

L'échangeur peut être facilement modélisé dans Thermoptim.

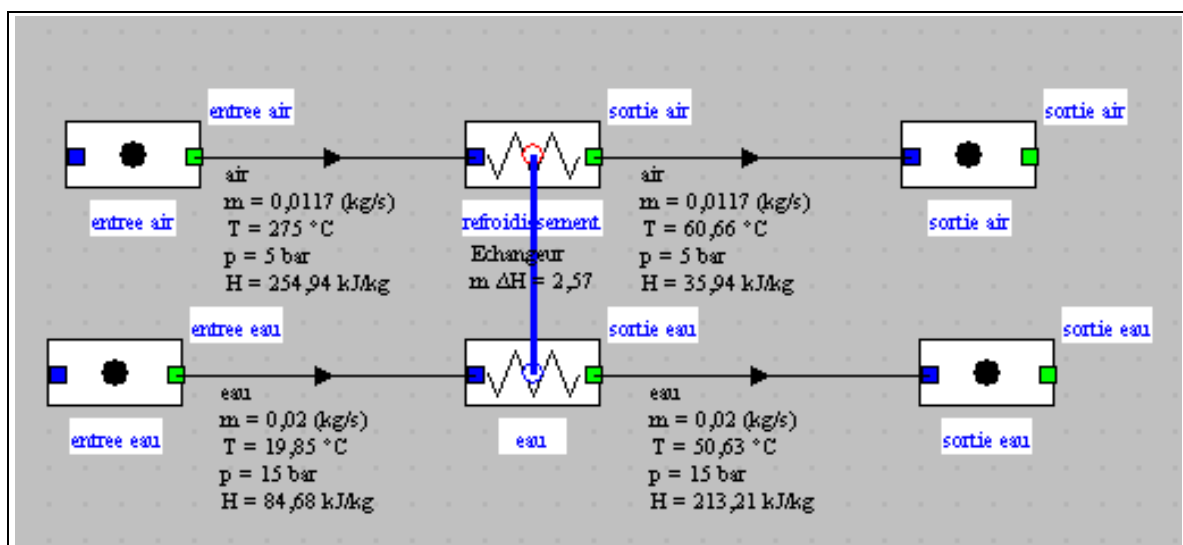


Figure 3.9 : Synoptique de l'exemple

Pour dimensionner l'échangeur, on se fixe une efficacité de 0,84.

En utilisant les écrans de dimensionnement technologique de Thermoptim, il est possible de calculer le coefficient d'échange thermique global et d'en déduire la surface nécessaire pour transmettre la puissance souhaitée.

Il faut pour cela déterminer les diamètres hydrauliques d_h et les sections de passage Ac des deux fluides.

A l'intérieur des tubes, d_h est donné, et le calcul de Ac est très simple : il est égal au produit du nombre de tubes par la section unitaire. A l'extérieur des tubes, les calculs sont un peu plus compliqués. d_h est égal à 4 fois la section de passage par le périmètre mouillé. Quant à Ac , c'est la surface transversale offerte à l'air.

On a supposé que les ailettes multipliaient par 4 la surface d'échange, avec une efficacité égale à 0,8.

Ces valeurs doivent être entrées dans l'écran de dimensionnement technologique de l'échangeur. Après quelques tâtonnements, on obtient un paramétrage proche de celui de la figure 3.11.

Le pilote peut alors calculer la surface nécessaire, qui est ici égale à $0,067 \text{ m}^2$, les coefficients d'échange valant $669.3 \text{ W/m}^2/\text{K}$ côté air, et 759.1 côté eau, le tout conduisant donc à un coefficient global égal à $355.7 \text{ W/m}^2/\text{K}$.

L'échangeur sera donc constitué de deux tubes de 90 cm disposés en serpentins sur 3 niveaux et traversant des plaques de tôle d'une surface étendue totale de $0,31 \text{ m}^2$, séparées les unes des autres de 3 mm.

nom: Echangeur type: contre-courant

fluide chaud: refroidissement afficher

fluide froid: eau afficher

Tce (°C): 275 imposé

Tcs (°C): 60,66190079 calculé

mc: 0,0117378 imposé

Cpc: 1,02174976

m ΔHc: -2,57057703

Tfe (°C): 19,85 imposé

Tfs (°C): 50,63221194 calculé

mf: 0,02 imposé

Cpf: 4,17543186

m ΔHf: 2,57058057

UA: 0,0238637478 dimensionnement

R: 0,143615214

NUT: 1,99012514

DTML: 107,70081491

non contraint

pincement minimum DTmin: 0

efficacité imposée epsilon: 0,84004742

Calculer

tech. design

Sauver

Supprimer

Fermer

Figure 3.10 : Ecran de l'échangeur

Echangeur

air

hlc
hlcvc
hlc = 669.34 Re = 229.38

hlf
hlfvc
hlf = 759.14 Re = 1442.18

ext_tube | Colburn correlation for single phase flow outside tubes

correlation settings

local ΔP loss coeff. 0

pressure drop 0.000384

friction factor 0.279019

refroidissement

eau

free flow area 0.0027

hydr. diameter 0.0013

length 0.06

surface factor 4

fin efficiency 0.8

free flow area 0.000226

hydr. diameter 0.012

length 0.9

surface factor 1

fin efficiency 1

int_tube | Mac Adams correlation for single phase flow inside tubes

correlation settings

local ΔP loss coeff. 0

pressure drop 0.000131

friction factor 0.044378

Quitter

Figure 3.11 : Ecran de dimensionnement technologique

Comme nous l'avons dit dans la section précédente, il serait ici possible d'utiliser le pilote générique pour construire cet écran technologique. Toutefois, cet exemple simple se prêtant bien à une première réalisation, nous présentons ci-dessous la manière d'en créer un.

3.3.2 Conception du pilote

La classe du pilote est appelée `PiloteEchangeurAir`. Son interface graphique étant simple et classique, nous ne la détaillerons pas ici.

3.3.2.1 Initialisations

Pour connaître le nom de l'échangeur, on pourrait utiliser la méthode `getHxList()` de `Projet`, mais elle impose que le schéma soit chargé dans l'éditeur. Si l'on veut éviter cette contrainte, il suffit de le déclarer directement dans le code.

Pour les autres initialisations, on utilise des instances de `PointThopt`, qui sont des sortes de clones des points du noyau de `Thermoptim`, et permettent d'avoir un accès aisé à leurs valeurs. Quatre de ces objets sont instanciés, représentant les entrées et sorties des deux fluides. Ici, nous avons entré leurs noms à la main, mais il serait possible de les reconnaître automatiquement.

```
//initialisations pour la simulation
//attention : les noms des points et composants doivent être exacts, sous
hxName="Echangeur";
amontChaud=new PointThopt(proj,"entree air");
avalChaud=new PointThopt(proj,"sortie air");
amontFroid=new PointThopt(proj,"entree eau");
avalFroid=new PointThopt(proj,"sortie eau");
```

L'écran technologique est ensuite instancié et le Vector `vTechno` construit :


```

technoEchangeur=new TechnoHx(proj, hxName, amontChaud, avalChaud, amontFroid, a
addTechnoVector(technoEchangeur);
setupTechnoDesigns(vTechno);

```

La dernière ligne permet d'initialiser le TechnoDesign lors de l'ouverture d'un fichier de projet existant ou lors du chargement initial du pilote. Sans elle, ce dernier n'apparaît pas dans la liste des TechnoDesign de l'écran d'affichage technologique et de simulation de la figure 3.5.

3.3.2.2 Calculs

```

if(!hxName.equals("")){//initialisation de l'évaporateur
    args=new String[2];
    args[0]="heatEx";
    args[1]=hxName;
    vProp=proj.getProperties(args);
    Double f=(Double)vProp.elementAt(15);
    UAech=f.doubleValue();

    amontChaud.getProperties();
    avalChaud.getProperties();
    amontFroid.getProperties();
    avalFroid.getProperties();

    //initialisations du TechnoDesign
    technoEchangeur.UA=UAech;
    technoEchangeur.makeDesign();

    //affichages
    U_value.setText(Util.aff_d(technoEchangeur.getU(),4));
    UAech_value.setText(Util.aff_d(UAech,4));
    AechReel=Util.lit_d(technoEchangeur.ADesign_value.getText());
    AcalculatedEch_value.setText(Util.aff_d(AechReel,4));
}

```

Les calculs sont ici très simples : en utilisant la méthode `getProperties()` de `Projet`, le pilote récupère les valeurs de UA et ΔH . L'échangeur est ensuite dimensionné par la méthode `makeDesign()` et les résultats sont affichés à l'écran.

3.3.2.3 Sauvegardes

Dans cet exemple, il n'y a pas de sauvegardes particulières à effectuer si l'on considère constantes les résistances thermiques d'encrassement. Les méthodes standard de sauvegarde des paramètres technologiques font l'affaire.

Deux méthodes d'`ExtPilot`, `transferTechnoData()` et `setupTechnoDesigns()`, permettent d'initialiser le pilote et les écrans technologiques lors de l'ouverture d'un fichier de projet existant.

La première méthode regroupe toutes les lignes du fichier de projet relatives aux écrans technologiques et les charge dans le pilote en attendant qu'il soit construit, et la seconde décode ces lignes pour effectuer ses initialisations, en passant le relais aux méthodes standard des `TechnoExch`, qui font quant à elles appel aux `FlowConfig`.

La dernière ligne de `setupTechnoDesigns()`, `proj.bindTechnoDesigns(vSettings)`, sert à relier les `TechnoDesign` utilisés par le pilote avec les composants du noyau, afin qu'ils puissent être affichés à partir de leurs écrans et des tables générales de l'écran d'affichage technologique et de simulation.

4 NON - NOMINAL

La structure informatique développée pour pouvoir effectuer des dimensionnements technologiques permet aussi d'analyser le comportement en régime non-nominal des modèles créés avec Thermoptim.

Ce type d'analyse présente cependant des difficultés complémentaires : il faut que le modélisateur soit capable de poser le jeu d'équations décrivant les couplages existants entre les différents composants du système étudié, et qu'il trouve un algorithme de résolution approprié.

Les cas que nous avons traités nous ont montré qu'il existe deux grandes difficultés complémentaires :

- la première concerne la mise en équation des phénomènes physiques. Il faut analyser avec soin et modéliser correctement le comportement des différents composants, ce qui peut être loin d'être simple. Nous proposons dans les exemples mis en ligne sur le portail Thermoptim-Unit un certain nombre de modèles, mais ils ne sont pas les seuls possibles, tout comme nous n'avons pas traité tous les cas intéressants ;
- la seconde correspond à la résolution, généralement numérique, des équations du modèle, ce qui demande d'une part de trouver un algorithme adéquat, et d'autre part de l'initialiser correctement.

Notre propos étant principalement centré sur la thermodynamique appliquée, nous nous sommes essentiellement concentrés sur la première difficulté. Les outils de résolution numérique que nous proposons utilisent des bibliothèques soit propres à Thermoptim, pour effectuer de simples recherches par dichotomie, soit génériques, comme Minpack, qui est un ensemble de classes permettant de faire de l'optimisation non linéaire, et en particulier de chercher la solution de systèmes d'équations, qui sera présenté dans la section 4.3.

Nous commencerons dans un premier temps par traiter un exemple simple qui ne pose pas de difficulté de résolution numérique et d'expliquer le code de la classe du pilote.

4.1 Exemple

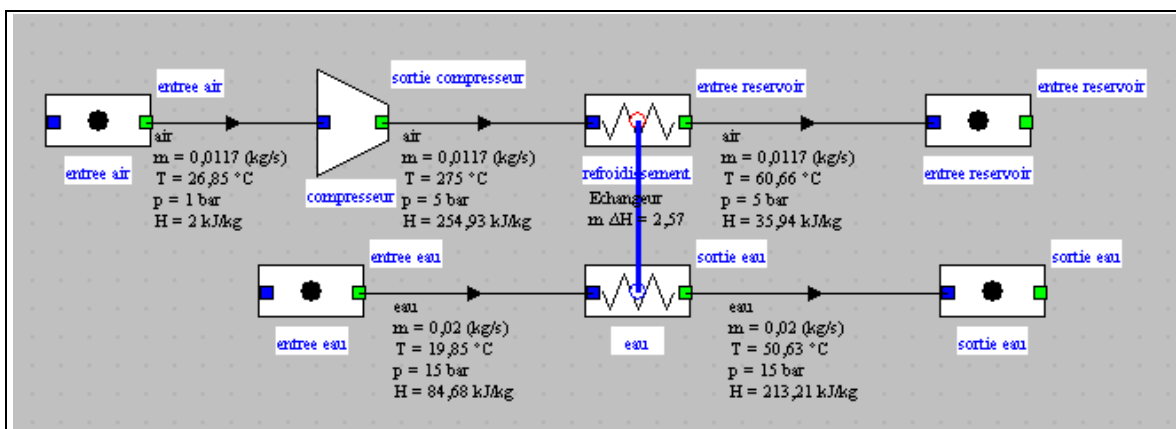


Figure 4.1 : Exemple de compresseur refroidi

Pour illustrer la capacité qu'a Thermoptim d'effectuer des calculs en régime non-nominal, nous allons étudier le comportement d'un compresseur volumétrique à air qui remplit un stockage d'air comprimé de volume donné à pression variable. L'air comprimé est refroidi avant stockage grâce à un échangeur à eau du type de celui qui vient d'être présenté.

Le système peut être facilement modélisé dans Thermoptim et conduit à un schéma du type de la figure 4.1, qui ne se distingue de celui étudié lors de l'exemple précédent que par l'ajout du compresseur.

L'écran de dimensionnement technologique du compresseur est donné figure 4.2. Celui de l'échangeur est similaire au précédent. Ils correspondent aux modèles physiques présentés section 3.1 de ce manuel.

Notez, car nous ne l'avons pas encore dit, qu'un double-clic sur le nom du composant (ici "compresseur" situé sous les petites flèches en haut à droite) permet d'accéder à son écran thermodynamique.

Figure 4.2 : Ecran du compresseur

4.1.1 Résultats obtenus

Une fois le pilote réalisé, il est très facile de faire varier le rapport de compression pour obtenir les évolutions des principales grandeurs lorsque la pression du réservoir varie. Elles peuvent être exploitées avec la macro Excel de post-traitement des fichiers de simulation de ThermoOptim présentée au tome 1 du manuel de référence. Il suffit pour cela de sauvegarder le fichier de projet sous un nom différent après chaque simulation, puis de charger ces fichiers dans la macro et d'en extraire les valeurs intéressantes.

La figure 4.3 montre, en fonction de la pression du stockage, les évolutions du rendement isentropique du compresseur et du travail consommé, le débit d'air aspiré, la température de l'air entrant dans le réservoir, la charge de l'échangeur et la valeur de U.

4.1.2 Conception du pilote

La classe du pilote est `PiloteCompresseurVs`, et son type est "Pilote compresseur Vs". Son interface graphique étant simple et classique, nous ne la détaillerons pas ici.

L'écran du pilote est présenté figure 4.4.

Pour réaliser la classe correspondante, on reprend celle du pilote de l'échangeur, et on la complète sur deux points :

- d'une part en instanciant et en initialisant l'écran technologique du compresseur, de manière analogue à celui de l'échangeur ;
- d'autre part en introduisant un bouton "Calculate", et en définissant les calculs et modifications du paramétrage du simulateur qu'il convient d'effectuer lorsqu'on fait varier la pression aval.

4.1.2.1 Initialisations

Lors de l'initialisation, après que les `PointThopt` et les deux `TechnoDesign` aient été construits, la cylindrée du compresseur permettant d'obtenir le débit souhaité est déterminée, sur la base du paramétrage de l'écran technologique.

Figure 4.4 : Ecran du pilote

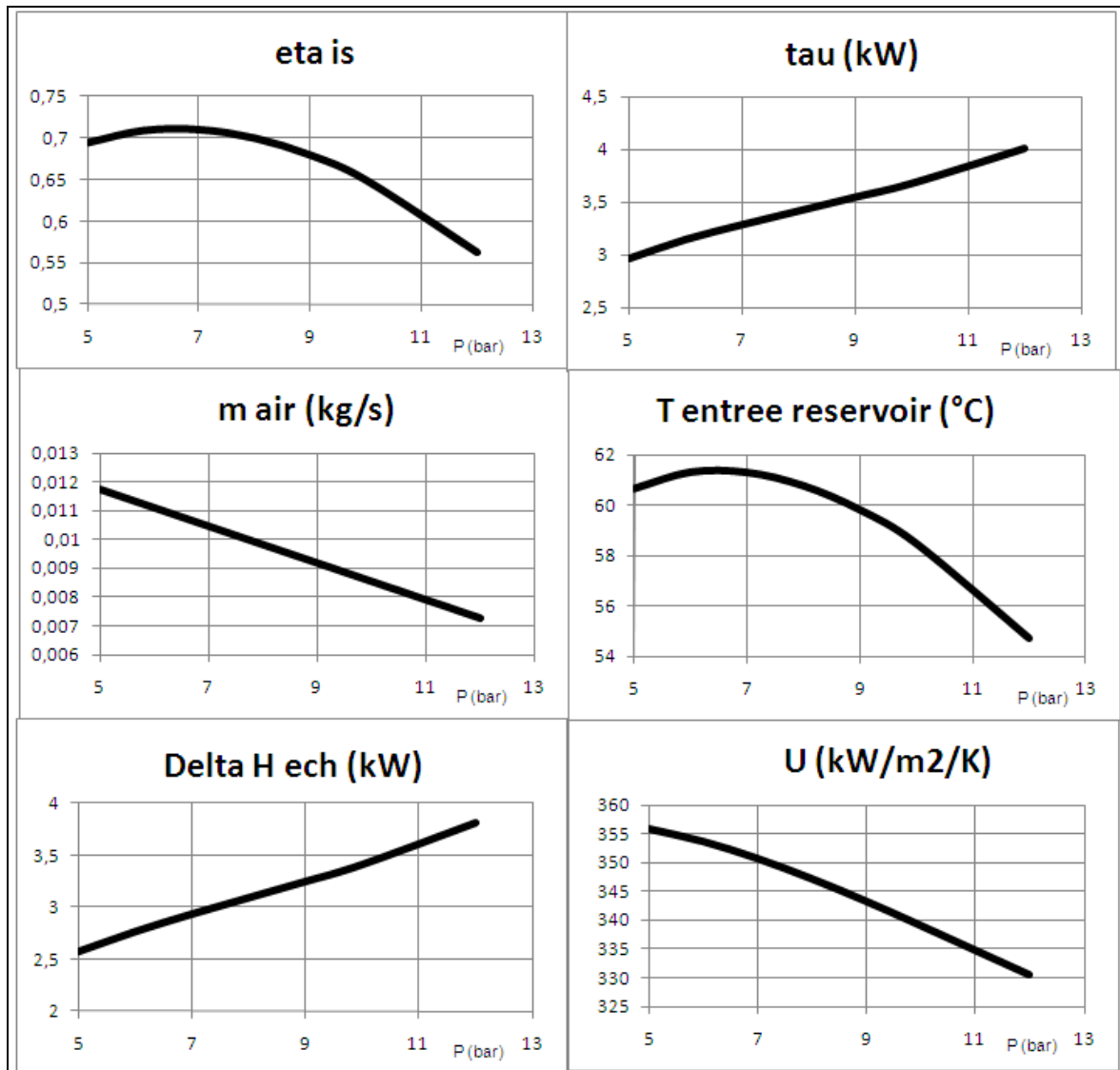


Figure 4.3 : Résultats de simulation

```
//initialisations des PointThopt et des TechnoDesign
//attention : les noms des points et composants doivent être exacts, sous peine de gén

hxName="Echangeur";
compressorName="compresseur";
amontChaud=new PointThopt(proj,"sortie compresseur");
avalChaud=new PointThopt(proj,"entree reservoir");
amontFroid=new PointThopt(proj,"entree eau");
avalFroid=new PointThopt(proj,"sortie eau");
amontCompr=new PointThopt(proj,"entree air");

//instanciation des TechnoDesign dans les classes externes
technoEchangeur=new TechnoHx(proj, hxName, amontChaud, avalChaud, amontFroid, avalFroi
addTechnoVector(technoEchangeur);
technoCompr=new VolumCompr(proj, compressorName, amontCompr, amontChaud);
addTechnoVector(technoCompr);

//initialisation des TechnoDesign dans ThermoOptim
setupTechnoDesigns(vTechno);
```

```

if(!compressorName.equals("")){//initialisation du compresseur
    args=new String[2];
    args[0]="process";
    args[1]=compressorName;
    vProp=proj.getProperties(args);
    String amont=(String)vProp.elementAt(1);
    String aval=(String)vProp.elementAt(2);
    Double f=(Double)vProp.elementAt(3);
    massFlow=f.doubleValue();
    N_value=Util.lit_d(technoCompr.N_value.getText());
    lambdaVol=technoCompr.getLambdaVol();
    Vs=massFlow*60*amontCompr.V/N_value/lambdaVol;
    Vs_value.setText(Util.aff_d(Vs,8));
    technoCompr.setVs(Vs);
    technoCompr.setN(N_value);
}

```

4.1.2.2 Calculs

Les calculs à effectuer sont les suivants ;

- commencer par initialiser la cylindrée et mettre à jour la pression de sortie du compresseur
- calculer les rendements volumétrique et isentropique, déterminer le débit massique mis en jeu et le propager en amont et en aval
- recalculer le compresseur et la transfo aval, sachant que celle-ci, paramétrée comme isobare, propage la nouvelle pression
- mettre à jour les entrées et sorties de l'échangeur puis le recalculer en non-nominal après avoir actualisé le débit calorifique de l'air qui a été modifié
- mettre à jour le simulateur et recalculer le projet plusieurs fois pour garantir la stabilisation des valeurs.

Cet exemple illustre encore mieux que le précédent l'intérêt qu'il y a à utiliser les PointThopt pour communiquer entre le pilote et le simulateur. La syntaxe des mises à jour est beaucoup plus lisible qu'en utilisant seulement les méthodes `getProperties()` et `updatePoint()` de `Projet`.

```

void bCalc_actionPerformed(java.awt.event.ActionEvent event)
{
    Vs=Util.lit_d(Vs_value.getText());//mise à jour de la cylindrée du compresseur
    technoCompr.setVs(Vs);
    Preservoir=Util.lit_d(P_value.getText());
    double UA_ech=Util.lit_d(UAech_value.getText());

    amontChaud.P=Preservoir;// mise à jour de la pression de sortie du compresseur
    amontChaud.update(!UPDATE_T,UPDATE_P,!UPDATE_X);
    amontChaud.getProperties();

    massFlow=technoCompr.getMassFlow(amontCompr.V);
    double eta_is=technoCompr.getRisentr();// calcul du rendement isentropique du compresseur
    lambdaVol=technoCompr.getLambdaVol();

    //recalcul du compresseur et de la transfo aval
    updateprocess(compressorName, "Compression",RECALCULATE,IS_SET_FLOW, UPDATE_FLOW, massFlow, U
    amontChaud.getProperties();
    updateprocess("refroidissement", "Exchange",RECALCULATE,IS_SET_FLOW, UPDATE_FLOW, massFlow, U
    updateprocess("entree air", "Exchange",RECALCULATE,IS_SET_FLOW, UPDATE_FLOW, massFlow, UPDATE

    amontChaud.getProperties();//mise à jour des entrées et sorties de l'échangeur
    avalChaud.getProperties();
    amontFroid.getProperties();
    avalFroid.getProperties();
}

```

Le calcul du compresseur se fait grâce aux deux méthodes `getRisentr()` et `getLambdaVol()` introduites précédemment. La méthode `updateprocess()` modifie le débit et le rendement isentropique du compresseur puis le recalcule. Son point aval, qui s'appelle ici "amontChaud" car il correspond à l'amont du fluide chaud de l'échangeur, est ensuite mis à jour.

Le calcul de l'échangeur est fait de la manière suivante : on fait un premier calcul avec la méthode `updateHx()` en imposant la valeur de UA lue à l'écran, UA_ech (l'échangeur est paramétré pour un calcul en mode non-nominal, afin que ce soit la valeur de UA qui soit prise en compte). Ce calcul permet d'initialiser l'échangeur pour les nouvelles conditions de fonctionnement.

On fait ensuite appel à la méthode `makeDesign()` du `TechnoDesign`, ce qui met à jour la valeur de U. Le véritable UA s'obtient en multipliant le nouveau U avec la valeur de A initiale (AechReel), ce qui permet de recalculer l'échangeur correctement.

Etant donné que U dépend des températures moyennes des fluides, et donc de celles de sortie, on itère cinq fois les calculs pour garantir une bonne stabilisation, en réinitialisant à chaque fois UA_ech.

```
//calcul de l'échangeur (on itère plusieurs fois)
for(int i=0;i<5;i++){
    updateHx(hxName, RECALCULATE, UPDATE_UA, UA_ech, !UPDATE_EPSI, 0, !UPDATE_DTMIN, 0, UPDATE_CALC_MODE, OFF_DESIGN_MODE);
    avalFroid.getProperties();
    avalChaud.getProperties();

    technoEchangeur.UA=UA_ech;//détermination de U
    technoEchangeur.makeDesign();
    double U=technoEchangeur.getU();

    UAech=U*AechReel/1000;//mise à jour de UA et recalcul de l'échangeur
    UAech_value.setText(Util.aff_d(UAech,4));
    updateHx(hxName, RECALCULATE, UPDATE_UA, UAech, !UPDATE_EPSI, 0, !UPDATE_DTMIN, 0, UPDATE_CALC_MODE, OFF_DESIGN_MODE);

    avalFroid.getProperties();
    avalChaud.getProperties();
    U_value.setText(Util.aff_d(U,4));
    UA_ech=UAech;
}

//mise à jour du simulateur et des affichages
for(int j=0;j<3;j++){proj.calcThopt();
    flow_value.setText(Util.aff_d(massFlow,4));
    eta_is_value.setText(Util.aff_d(eta_is,4));
    lambdaVol_value.setText(Util.aff_d(lambdaVol,4));
    AcalculatedEch_value.setText(technoEchangeur.ADesign_value.getText());
}
```

4.1.2.3 Sauvegardes

Les sauvegardes et mises à jour lors des lectures de fichier de projet sont analogues à celles de la classe `PiloteEchangeur`. La méthode `setupPilot()` a été complétée pour tenir compte de l'existence du compresseur.

4.2 Utilisation du modèle pour simuler le remplissage d'un stockage d'air comprimé

Les résultats obtenus peuvent être utilisés pour simuler le remplissage d'un stockage d'air comprimé. Pour cette simulation, nous développons un second modèle, résolu sous Excel, avec les équations suivantes :

- la masse M d'air contenue dans le stockage est égale à la masse initiale plus l'intégrale du débit
- l'énergie interne U est égale à l'énergie interne initiale plus l'intégrale du produit du débit par l'enthalpie de l'air sortant du refroidisseur, moins l'intégrale des pertes par convection avec l'air ambiant
- la température du stockage est déduite de son énergie interne
- la pression se calcule alors par la loi des gaz idéaux.

Pour connaître le débit, le travail de compression et la température de l'air en entrée de stockage, il suffit de faire des régressions polynomiales à partir des simulations effectuées précédemment avec `Thermoptim`. Les équations obtenues sont, en fonction du rapport de compression r :

$$\dot{m} = -0,0006 r + 0,0149$$

$$\tau = 0,0019 r^3 - 0,0484 r^2 + 0,5322 r + 1,2806$$

$$T_{air} = 0,01 r^3 - 0,4735 r^2 + 4,9142 r + 46,699$$

L'allure des résultats est donnée ci-dessous, pour un stockage d'un demi m³ (diamètre 80 cm, longueur 1 m), alimenté par un compresseur de cylindrée $V_s = 0,525$ l. La pression et la température dans le stockage, sa masse et le travail à fournir au compresseur sont affichés figure 4.5.

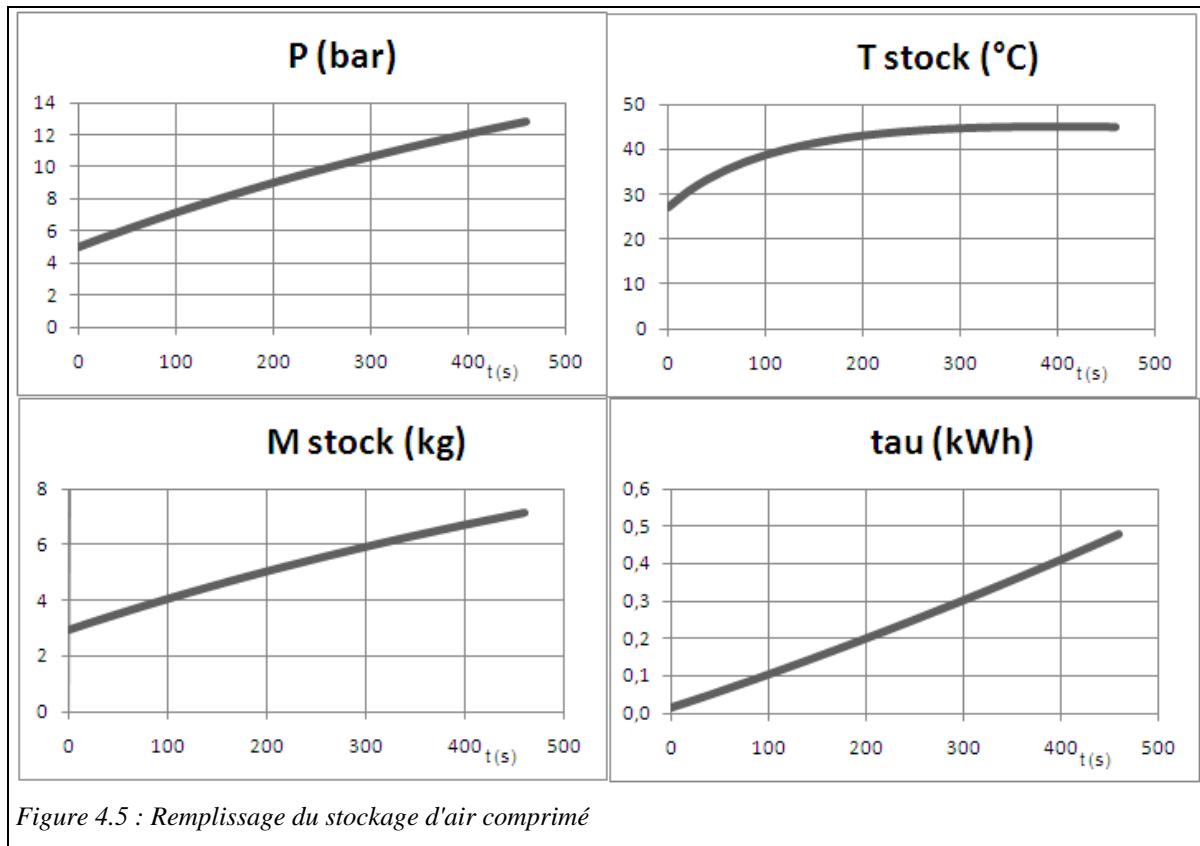


Figure 4.5 : Remplissage du stockage d'air comprimé

Cet exemple vous a permis de faire l'apprentissage de l'écriture d'un pilote destiné à simuler le comportement en régime non-nominal d'un système thermodynamique relativement simple. Pour approfondir la question, nous vous conseillons d'étudier la notice de programmation d'un pilote de machine frigorifique disponible sur le portail Thermoptim-UNIT², qui, compte tenu du nombre d'équations couplées qu'il met en jeu, nécessite l'utilisation d'un solveur.

4.3 Utilisation du solveur minPack pour résoudre des systèmes d'équations non-linéaires

Dans de nombreux cas, les équations décrivant le comportement non-nominal d'un système énergétique sont non-linéaires, de telle sorte que leur résolution constitue un problème difficile, surtout lorsque le nombre d'inconnues est élevé.

4.3.1 Présentation de minPack

Basé sur la méthode de Levenberg-Marquardt, minPack est un ensemble d'algorithmes mis au point en Fortran, puis traduits sous Java. Cette méthode combine l'algorithme de Gauss-Newton et la méthode de descente de gradient. Son intérêt principal est d'être très robuste et de ne nécessiter comme initialisation qu'une solution approchée.

Précisons que minPack est en fait un ensemble d'algorithmes de minimisation qui peut être utilisé pour trouver le minimum de la norme L2 des résidus $\|F(X)\|$ d'un ensemble de m équations non-linéaires à n inconnues $F(X)$.

² <http://www.thermoptim.org/sections/logiciels/thermoptim/documentation/pilote-pour-cycle>

Lorsque n est égal à m , minPack peut être utilisé pour résoudre le système d'équations $F(X) = 0$, ou tout au moins chercher un vecteur X qui soit la plus proche possible de la solution.

On peut aussi utiliser minPack pour identifier un jeu de n paramètres permettant d'ajuster une équation non linéaire sur un ensemble de m données.

4.3.2 Mise en œuvre de minPack

L'ensemble des classes nécessaire est inclus dans la bibliothèque extBib.zip qui doit être déclarée dans le classPath.

L'implémentation sous Java de minPack se fait en utilisant une interface appelée optimization.Lmdif_fcn, qui contraint les classes appelantes à disposer d'une fonction appelée fcn().

Cette fonction fcn() reçoit comme principaux arguments un vecteur (tableau $x[n]$)³ contenant les variables et un vecteur (tableau fvec[m]) renvoyant les résidus des fonctions que l'on cherche à annuler. Comme nous l'avons indiqué, leur nombre peut excéder celui des variables, mais il doit être le même si on cherche la solution d'un système d'équations.

Le guidage de l'algorithme se fait en pratique en jouant sur deux critères de précision, l'un portant sur la somme des résidus, et l'autre sur la précision du calcul des dérivées partielles, estimées par différences finies.

4.3.3 Exemple

Pour illustrer l'emploi de minPack, nous avons réalisé une classe de test, appelée TestMinPack, qui résout le système d'équations correspondant à l'intersection d'un cercle et d'une droite. Le code se présente comme suit :

```
int info[] = new int[2];
int m = 2;//système de deux équations
int n = 2;//à deux inconnues
int iflag[] = new int[2];
double fvec[] = new double[m+1];//vecteur des résidus
double x[] = new double[n+1];
double residu0;//norme L2 initiale
double residu1;//norme L2 finale

System.out.print("\n\n\n\n problem dimensions: " + n + " " + m + "\n");

//initialisation des inconnues
x[1] = 1;
x[2] = 0;

iflag[1] = 0;

testMinPack.fcn(m,n,x,fvec,iflag);//premier appel pour calcul norme L2 initiale
//norme L2 initiale
residu0 = optimization.Minpack_f77.enorm_f77(m,fvec);

testMinPack.nfev = 0;
testMinPack.njev = 0;

double epsi=0.0005;//précision globale demandée sur la convergence
double epsfcn = 1.e-6;//précision calcul des différences finies
```

³ Attention : afin de conserver les mêmes indices qu'en Fortran, l'implémentation Java déclare des tableaux de dimension $n+1$ au lieu de n , l'indice 0 n'étant pas utilisé


```
//appel à minPack
optimization.Minpack_f77.lmdif2_f77(testMinPack, m, n, x, fvec, epsi, epsfcn, info);

//norme L2 finale
residu1 = optimization.Minpack_f77.enorm_f77(m,fvec);

//affichage des résultats
System.out.println("\n Initial L2 norm of the residuals: " + residu0 +
    "\n Final L2 norm of the residuals: " + residu1 +
    "\n Number of function evaluations: " + testMinPack.nfev +
    "\n Number of Jacobian evaluations: " + testMinPack.njev +
    "\n Info value: " + info[1] );
System.out.println();
System.out.println("precision: \t" + residu1);
for(int i=1;i<=n;i++){//affichage de la solution
    System.out.println("x["+i+"] \t" + x[i]);
}
```

La fonction fcn est définie comme suit :

```
public void fcn(int m, int n, double x[], double fvec[], int iflag[]) {

    if (iflag[1]==1) this.nfev++;
    if (iflag[1]==2) this.njev++;

    fvec[1] =x[1]*x[1]+x[2]*x[2]-1;
    fvec[2] =3*x[1]+2*x[2]-1;
}
```

Les résultats que l'on obtient sont les suivants, pour l'initialisation $x[1] = 1$, $x[2] = 0$:

problem dimensions: 2 2

```
Initial L2 norm of the residuals: 2.0
Final L2 norm of the residuals: 2.39665398638067E-10
Number of function evaluations: 6
Number of Jacobian evaluations: 10
Info value: 2
```

```
precision:      2.39665398638067E-10
x[1]    0.7637079407212384
x[2]   -0.6455619110818576
```

pour l'initialisation $x[1] = 0$, $x[2] = 0$:

problem dimensions: 2 2

```
Initial L2 norm of the residuals: 1.4142135623730951
Final L2 norm of the residuals: 3.853333208070353E-10
Number of function evaluations: 9
Number of Jacobian evaluations: 12
Info value: 2
```

```
precision:      3.853333208070353E-10
x[1]   -0.3021694793631984
x[2]    0.9532542190447976
```

Cet exemple illustre d'une part comment minPack peut être utilisé, mais aussi l'importance qu'il y a à bien initialiser le problème soumis. Dans notre cas, le problème a deux solutions, mais l'algorithme se satisfait de la première qu'il trouve, la plus proche de la valeur de départ.

5 METHODOLOGIE D'IDENTIFICATION DES PARAMETRES DE COMPOSANTS

Lorsque l'on cherche à effectuer des calculs en régime non-nominal, deux des principales difficultés méthodologiques sont l'obtention de données expérimentales exploitables et la paramétrisation des modèles de composants retenus. Nous supposons dans ce qui suit que l'on dispose de données mesurées bien validées.

L'expérience prouve d'ailleurs que les méthodes qui sont présentées ici permettent de diagnostiquer des incohérences entre les données s'il en existe.

Cette section à caractère méthodologique traite de l'identification des paramètres des composants. La problématique à laquelle il s'adresse étant en effet particulièrement difficile à présenter dans l'abstrait, nous avons jugé préférable de l'illustrer par des exemples précis, en détaillant la manière de parvenir aux résultats, ce qui suppose d'expliquer comment opérer en pratique.

Pour mener à bien cette démarche délicate, Thermoptim a été doté d'écrans particuliers accessibles depuis le menu "Dimensionnement technologique" de l'écran du simulateur. Ces écrans permettent :

- d'importer des données expérimentales dans des fichiers de projet ;
- d'automatiser des simulations portant sur des séries de fichiers de projet ;
- d'effectuer des analyses de sensibilité sur l'influence des paramètres qui interviennent dans les corrélations utilisées pour dimensionner les échangeurs.

Il faut en effet dans un premier temps pouvoir saisir les données expérimentales disponibles sous forme de fichiers de projet de Thermoptim. En règle générale, il est préférable de commencer par identifier les paramètres des différents composants isolément, pour ensuite étudier le comportement du système couplé. Les procédures présentées ici permettent d'automatiser la création de fichiers de projet relatifs à des condenseurs et des évaporateurs, ainsi qu'à des compresseurs volumétriques.

Une fois que l'on a généré un ensemble de fichiers de projet de Thermoptim conforme aux données expérimentales, le processus d'identification consiste à essayer plusieurs jeux de valeurs des paramètres de dimensionnement des composants, pour en trouver un qui soit cohérent compte tenu à la fois des valeurs mesurées et de ce que l'on connaît de la géométrie et de la technologie des appareils testés. Pour faciliter ce processus, deux familles d'écrans ont été créées. La première permet d'effectuer de manière générique des modifications des fichiers de projet et de lancer des séries de simulations, qui peuvent ensuite être exploitées en utilisant la macro de post-traitement des fichiers de Thermoptim. La seconde est spécifique au paramétrage des échangeurs.

5.1 Importation de données expérimentales

Il faut commencer par créer un fichier de projet du composant, en le paramétrant aussi bien que possible sur le plan géométrique, et on aura généralement aussi intérêt à créer le schéma correspondant. Comme indiqué ci-dessous, le paramétrage sera effectué sur la base des noms des points et transfos. Pour faciliter la comparaison avec les résultats expérimentaux, il est préférable d'adopter les mêmes conventions dans le modèle Thermoptim que sur le banc d'essais.

La ligne "Importer des données expérimentales" du menu "Dimensionnement technologique" affiche un écran qui permet d'importer dans un fichier de projet de Thermoptim des données expérimentales (figure 5.1.1). Cet écran donne accès à trois menus et boutons, dont les fonctions sont équivalentes.

Le principe est le même pour tous les boutons :

- un projet doit avoir été chargé préalablement à l'importation des données ;

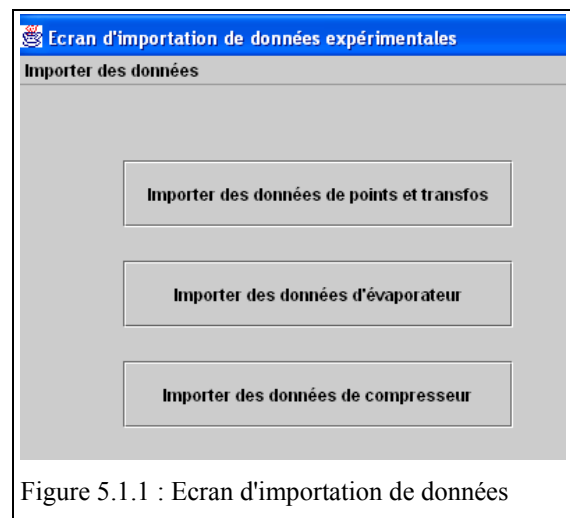


Figure 5.1.1 : Ecran d'importation de données

- un fichier de données convenablement structuré (figure 5.1.2) est relu. Chaque ligne contient les valeurs d'une part des températures et pressions de différents points, et d'autre part des débits de différentes transfos du projet. Lors de la lecture d'une ligne, les points et transfos sont mis à jour, et des calculs spécifiques sont éventuellement effectués ;
- le fichier de projet est enfin sauvé sous un nouveau nom dans le répertoire "ModifiedProjects", et la ligne suivante est lue.

La figure 5.1.2 montre la structure d'un fichier de données par défaut, correspondant au bouton "Import data for point and processes" :

- la première ligne n'est pas relue ;
- la seconde contient en deuxième colonne le nombre de valeurs à relire ;
- la troisième indique le nombre de points (n_p) ;
- la quatrième donne le nombre de transfos (n_t) ;
- la cinquième n'est pas relue : elle permet si on le désire de documenter le contenu des différentes colonnes ;
- la sixième permet à ThermoOptim de reconnaître les nom des points et des transfos qui devront être modifiés ;
- les lignes suivantes sont les lignes de données.

	A	B	C	D	E	F	G	H	I	J	K
1	Fichier de paramétrage de projets ThermoOptim										
2	valeurs	80									
3	points	4									
4	transfos	2									
5		Tin caloporteur	Pin caloporteur	Tout caloporteur	Pout caloporteur	Tin réfrigérant	Pin réfrigérant	Tout réfrigérant	Pout réfrigérant	débit caloporteur	débit réfrigérant
6	libellés	tgec	tgec	tgsc	tgsc	tfec	tfec	tfsc	tfsc	tgec	tfec
7	1	17.61	1	30.7	1	52.87	15.08	25.81	15.08	0.1	0.0336
8	2	19.5	1	30.63	1	52.47	15.07	27.6	15.07	0.11	0.0315
9	3	18.21	1	30.64	1	55.04	15.08	24.99	15.08	0.11	0.0339
10	4	18.74	1	30.11	1	54.11	14.8	25.7	14.8	0.11	0.032
11	5	17.11	1	29.75	1	57.87	14.88	24.85	14.88	0.11	0.0333
12	6	17.11	1	29.33	1	52.26	14.61	23.73	14.61	0.11	0.0327
13	7	18.16	1	28.69	1	52.71	14.31	23.02	14.31	0.11	0.029

Figure 5.1.2 : fichier d'importation de données expérimentales

La structure des lignes de données est la suivante :

- la première colonne contient la chaîne de caractères qui, lors de la sauvegarde des données importées, sera concaténée au nom du projet initialement chargé ;
- les n_p séries de deux colonnes suivantes fournissent les valeurs des températures (en °C) et des pressions (en bars) des différents points à importer (le nom du point à modifier est celui qui apparaît même colonne à la ligne 6 ; si aucun point de ce nom n'existe dans le projet, l'importation n'a pas lieu) ;
- les n_t séries de deux colonnes suivantes fournissent les valeurs des débits des différentes transfos à importer (le nom de la transfo à modifier est celui qui apparaît même colonne à la ligne 6 ; si aucune transfo de ce nom n'existe dans le projet, l'importation n'a pas lieu).

Dans l'exemple de la figure 5.1.2, après lecture de la ligne 7, le point "tgec" est mis à la température de 17,61 °C, et à la pression de 1 bar, et le débit de la transfo "tgec" vaut 0,1 kg/s. Si le nom du fichier de projet initialement chargé est "monproj_.prj", le fichier de projet correspondant aura comme nom "monproj_1.prj".

5.1.1 Cas particulier d'un évaporateur

La plupart du temps, lorsqu'on effectue des mesures sur un évaporateur, on connaît les pressions et les températures d'entrée et de sortie des deux fluides, ainsi que leurs débits, mais on ignore le titre du réfrigérant en entrée. Le bilan enthalpique de l'échangeur permet cependant de l'estimer. Le bouton et le menu "Import evaporator data" permettent, comme les précédents, de paramétrer des fichiers de projet, avec en plus calcul du titre d'entrée et mise à jour du facteur de pression, utilisé pour prendre en compte une perte de charge.

Pour éviter toute confusion, le fichier d'importation doit avoir une structure particulière (figure 5.1.3) : il ne doit comporter que deux transfos, celle du réfrigérant ("refrig"), et celle du caloporteur ("mg1"), et les quatre points d'entrée et de sortie de ces deux transfos (dans un ordre indifférent, mais apparaissant avant les transfos dans les lignes du fichier). La valeur du facteur de pression est sauvée dans le TechnoDesign du réfrigérant : si l'on n'en veut pas, il faut soit entrer la même valeur des pressions en entrée et sortie dans le fichier de données, mais ceci induit un petit biais, soit charger le facteur de pression, puis le mettre à 1 avant simulation en utilisant la procédure de modification des fichiers de projet expliquée section 5.2.1.

Lors de la génération des fichiers de projet (qui sont sauvegardés dans le répertoire "ModifiedProjects"), un fichier texte appelé "evap.txt" est créé. Il contient, dans l'ordre des données du fichier d'importation, les valeurs de la température d'ébullition à la pression d'entrée du réfrigérant, du titre, et de la puissance frigorifique.

	A	B	C	D	E	F	G	H	I	J	K
1	Fichier de paramétrage de projets Thermoptim										
2	valeurs	80									
3	points	4									
4	transfos	2									
5		Tout détenteur				Pout détenteur				débit caloport	
6	libellés	tfee	tfee	tfee	tfee	tfee	tfee	tfee	tfee	mg1	refrig
7	1	25.81	4.81	-5.57	4.12	1.28	4.81	-1.41	4.12	0.43	0.0338
8	2	27.6	4.63	-6.63	3.97	1.39	4.63	-1.85	3.97	0.33	0.0315
9	3	24.99	4.59	-7.59	3.83	1.71	4.59	-1.82	3.83	0.33	0.0339
10	4	25.7	4.62	-6.92	3.96	1.4	4.62	-1.79	3.96	0.33	0.032
11	5	24.85	4.63	-1.58	3.85	1.32	4.63	-1.1	3.85	0.49	0.0333
12	6	23.73	4.7	-7.21	4.03	1.41	4.7	-1.64	4.03	0.38	0.0327
13	7	23.02	4.22	-10.67	3.61	1.59	4.22	-2.68	3.61	0.23	0.029

Figure 5.1.3 : fichier d'importation d'un évaporateur

5.1.2 Cas particulier d'un compresseur volumétrique

L'identification des paramètres d'un compresseur volumétrique présente différentes difficultés méthodologiques, tant pour le rendement volumétrique que pour le rendement isentropique, du fait que leur détermination est basée d'une part sur les mesures, et d'autre part sur les propriétés du réfrigérant. De plus, le fonctionnement en régime non-nominal fait appel à la vitesse de rotation du compresseur. On se reportera au chapitre 14 et à la section 5.4 pour de plus amples explications.

Le fichier d'importation doit donc avoir une structure particulière (figure 5.1.4) :

- les deux premiers points doivent correspondre, dans l'ordre, à l'aspiration et au refoulement ;
- à la différence du fichier par défaut, il y a trois colonnes par transfo, la première donnant le débit, la seconde la vitesse de rotation, et la troisième la puissance thermique mise en jeu (elle doit contenir des 0 pour un compresseur adiabatique). Pour les transfos qui ne sont pas des compressions, les seconde et troisième colonnes sont ignorées, ce qui explique que, dans la figure 5.5, les colonnes G et H soient identiques à la colonne F.

Lors de la génération des fichiers de projet (qui sont sauvegardés dans le répertoire "ModifiedProjects"), un fichier texte appelé "vsTs.txt" est créé (figure 5.1.5). Il contient les valeurs du volume massique et de l'enthalpie à l'aspiration et au refoulement, ainsi que celles de la température et de l'enthalpie de compression isenthalpique et de la température de refoulement calculée en régime non-nominal. A partir de ce fichier et de celui des données, il est possible de déterminer le rendement volumétrique, le rendement isentropique, et l'écart entre les températures de refoulement mesurée et calculée.

5.2 Automatisation des simulations de séries de fichiers de projet

	A	B	C	D	E	F	G	H	I	J	K
1	Fichier de paramétrage de projets Thermoptim										
2	valeurs	18									
3	points	2									
4	transfos	2									
5		Tout détenteur				Pout détenteur				débit caloport	
6	libellés	Tasp	Tasp	Tref	Tref	Tasp	Tasp	Tasp	compr	compr	compr
7	1	-1.506	2.795	107.8	12.74	0.0214	0.0214	0.0214	0.0214	870	-1.5905511
8	2	0.853	2.662	99.85	10.93	0.0209	0.0209	0.0209	0.0209	870	-1.7058897
9	3	-0.048	2.492	93.91	9.691	0.0198	0.0198	0.0198	0.0198	870	-1.5620094
10	7	0.807	2.584	105.7	10.8	0.0254	0.0254	0.0254	0.0254	1160	-1.93347
11	8	-2.245	2.481	108.2	11.15	0.0268	0.0268	0.0268	0.0268	1160	-1.9114569
12	0	-1.770	2.582	117.2	10.84	0.0274	0.0274	0.0274	0.0274	1160	-1.9100763

Figure 5.1.4 : fichier d'importation d'un compresseur

	A	B	C	D	E	F	G
1	Tasp_v	Tasp_h	Tref_v	Tref_h	Ts	hs	Tcalc
2	0.45797932	1475.24255	0.13797971	1695.4099	111.726796	1704.99318	111.583315
3	0.48631895	1482.59965	0.15819554	1681.53583	106.129125	1696.67195	100.466767
4	0.51875286	1482.005	0.17618668	1671.12065	100.315154	1686.42286	94.3295835
5	0.50140731	1483.23648	0.16317759	1696.02071	107.663687	1700.74184	106.056999
6	0.51653729	1476.64473	0.15902574	1701.01936	109.651542	1704.51691	108.859823

Figure 5.1.5 : fichier "vsTs.txt"

Identifier les paramètres d'un projet Thermoptim conduisant à la meilleure reconstitution de valeurs expérimentales consiste à chercher les valeurs de ces paramètres permettant, lors des simulations, d'obtenir l'écart le plus faible entre les résultats numériques et les mesures et les données technologiques. On commence pour cela, grâce aux procédures d'importation de données présentées section 5.1, par construire autant de fichiers de projet qu'il existe de relevés expérimentaux, en imposant les variables d'entrée telles que les pressions et températures en amont des composants.

Une fois ce jeu de fichiers de projets créé, le processus d'identification consiste à modifier, pour chacun d'eux, un ou plusieurs paramètres, comme par exemple un diamètre hydraulique ou une section de passage dans le cas d'un échangeur, puis à les simuler avec Thermoptim, en enfin à comparer les résultats obtenus avec les relevés expérimentaux.

Modifier successivement à la main ces paramètres devenant très vite fastidieux, il est nécessaire d'automatiser ce processus. La ligne "Automatic simulations" donne accès à l'écran de la figure 5.2.1, qui possède deux boutons pour lancer des actions, et un bouton d'arrêt en cas de problème.

Le bouton "Modifier et sauvegarder" permet d'effectuer une série de modifications sur un jeu de fichiers de projets placés dans le répertoire "ProjectsToModify", et de les sauver dans le répertoire "ModifiedProjects". Les modifications à effectuer sont définies dans un fichier présenté section 5.2.2.

Une fois ce jeu de fichiers de projets modifié, le bouton "Simuler" permet de les simuler, et de les sauver dans le répertoire "SimulatedProjects".

Les deux opérations présentées dans cette section sont donc automatiquement chaînées pour faciliter les traitements. Si l'on désire sauvegarder les fichiers modifiés ou simulés pour garder trace des opérations réalisées, il suffit de les compacter dans des archives, éventuellement avec un fichier de commentaires.

5.2.1 Modification des fichiers de projet

Ces rappels étant faits, expliquons comment fonctionne le bouton "Modifier et sauvegarder".

Par analogie avec le tableur de post-traitement, un fichier d'initialisation appelé "modifProj.ini" et structuré de la manière suivante, permet d'indiquer les modifications à effectuer (figure 5.2.1).

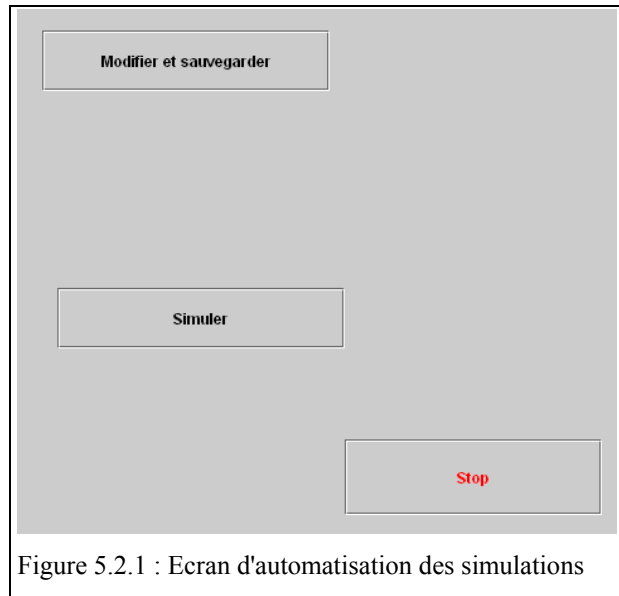


Figure 5.2.1 : Ecran d'automatisation des simulations

Les deux premières et la quatrième lignes sont des commentaires, mais elles ne doivent pas être vides.

La troisième ligne contient, en deuxième colonne, le nombre de valeurs à modifier.

A partir de la cinquième ligne apparaissent des quintuplets comportant :

- le nom de la grandeur à modifier (commentaire pour mémoire non utilisé par Thermoptim, mais qui ne doit pas être vide) ;
- la référence type tableur de cette grandeur dans le fichier de projet ;
- la nouvelle valeur à appliquer si le booléen de la cinquième colonne vaut "false" ;
- le facteur multiplicatif qui sera appliqué à la valeur existante si le booléen de la cinquième colonne vaut "true" ;
- la valeur du booléen permettant de choisir entre l'application à tous les fichiers d'une même valeur ou la multiplication des valeurs existantes par le même facteur.

Le séparateur des colonnes est la tabulation.

Sur le plan pratique, il faut commencer par placer le jeu de fichiers à modifier dans le répertoire "ProjectsToModify", les fichiers modifiés étant relus dans ce répertoire, puis modifiés et sauves sous le même nom dans le répertoire "ModifiedProjects".

5.2.2 Simulation d'une série de fichiers de projet

	A	B	C	D	E
1	fichier de modification des valeurs d'un projet Thermoptim				
2	entrez le nombre de valeurs à modifier, suivi des quintuplets				
3	nombre de valeurs	6			
4	grandeur	référence	valeur	facteur	multiplier
5	surface	B29	35	1.15	true
6	surface	D59	65	2	true
7	Dh	D57	65	2	true
8	DP	J57	3.33	2	false
9	Ac	E57	65	2	true
10	N	B51	35	1.5	true
44					

Figure 5.2.1 : structure de modifProj.ini

Le principe est ici très simple : tous les fichiers placés dans le répertoire "ModifiedProjects" sont successivement chargés et simulés dans l'écran du TechnoPilot jusqu'à soit convergence, soit atteinte de la limite de 30 itérations, soit message d'erreur conduisant à arrêter les calculs. Ils sont ensuite sauvés sous le même nom dans le répertoire "SimulatedProjects".

Cette méthode est relativement simple à mettre en œuvre puisqu'elle opère sur les seuls fichiers de projet, qui sont relus ensuite par Thermoptim et simulés. Son inconvénient est qu'elle ne permet pas d'effectuer des changements significatifs des valeurs à modifier, faute de quoi le progiciel risque de ne pas converger.

5.3 Analyse de l'influence des paramètres des corrélations des échangeurs

En pratique, on connaît très rarement suffisamment bien la géométrie interne de l'échangeur pour pouvoir estimer avec précision les paramètres que nous venons de présenter. Une seconde approche consiste à identifier les paramètres de l'échangeur à partir de données expérimentales. Afin de faciliter cette opération, des fonctionnalités spécifiques ont été implémentées dans Thermoptim pour permettre d'effectuer une série de dimensionnements de calcul de la surface A en faisant varier Ac et dh , pour un ensemble de fichiers de projet correspondant à des points d'essais.

5.3.1 Méthodologie de l'identification

Il est parfois difficile d'initialiser le paramétrage d'un échangeur multizones. Il faut commencer par bien étudier la configuration géométrique de l'échangeur, pour en déduire d'une part le meilleur choix parmi les configurations disponibles dans Thermoptim, et d'autre part des valeurs approchées des valeurs des paramètres requis.

Si, malgré ces précautions, le dimensionnement ne converge pas, on pourra commencer par bien équilibrer sur le plan enthalpique l'échangeur dans sa globalité, en calculant chaque projet à la main sans recourir au dimensionnement technologique. Attention dans ce cas à bien choisir les contraintes. Les algorithmes utilisés pour ce type d'échangeur étant beaucoup plus simples que pour les multizones, le calcul se fait en principe sans difficulté. Une fois cette opération effectuée, recommencez le dimensionnement.

Le titre de réfrigérant à l'entrée de l'évaporateur étant inconnu, il faut parfois ajouter dans le projet un laminage isenthalpique en amont de l'évaporateur, initialisé avec la pression et la température à l'entrée du détendeur.

On réalise le paramétrage en introduisant, dans l'écran de dimensionnement technologique, les caractéristiques géométriques et les valeurs des coefficients des corrélations utilisées pour chaque fluide.

En suivant cette procédure, on arrive à des points modélisés très proches des points mesurés, les seules différences étant :

- la pression de réfrigérant à la sortie de l'échangeur, la perte de charge n'étant pas prise en compte ;
- la température à l'entrée de l'évaporateur, légèrement modifiée du fait du calcul de la détente.

Une fois un point défini, on réalise un dimensionnement de l'échangeur à partir de l'écran technologique, ce qui fournit une valeur de la surface nécessaire. On obtient ainsi un jeu de valeurs de la surface de dimensionnement.

Notez bien que, pour pouvoir faire ce dimensionnement, il est nécessaire de **bâtir un pilote**, qui construit l'écran technologique de l'échangeur. Pour définir la précision de l'estimation des paramètres du modèle, il faut aussi se donner un critère. Celui que nous avons choisi est la dispersion des valeurs des surfaces de dimensionnement calculées pour les différents points de mesure.

On compare ainsi l'ensemble des simulations réalisées pour les jeux de paramètres retenus, ce qui conduit à réaliser de nombreuses simulations. Pour faciliter cette tâche, un mécanisme particulier a été introduit.

5.3.2 Automatisation des simulations

La ligne "Paramétrage des échangeurs de chaleur" du menu "Dimensionnement technologique" ouvre l'écran de la figure 5.3.1.

Il comprend dans la partie haute les valeurs des paramètres de la corrélation à utiliser. Les modifications des paramètres de la corrélation ne sont prises en compte que si l'option " modifier les paramètres de la corrélation de Nusselt " est choisie. Sinon, comme sur la figure 5.3.1, les composants du haut de l'écran sont désactivés.

Pour les zones diphasiques des échangeurs à plaques multizones, si l'option "used for refrigerant" est choisie, la corrélation s'applique aussi bien aux zones monophasiques côté réfrigérant qu'au caloporteur, sinon elle ne s'applique qu'au caloporteur, le réfrigérant utilisant son propre paramétrage.

Dans la partie basse de l'écran sont indiqués les paramètres de dimensionnement de l'échangeur sur lesquels les itérations doivent être réalisées : Ac et dh, avec leurs valeurs minimales et maximales et le nombre d'itérations à effectuer dans chaque cas (les incréments suivent une raison géométrique, et, si le nombre d'incrément demandé est égal à 1, seule la valeur minimale est considérée).

Figure 5.3.1 : Ecran de paramétrage de l'échangeur

De plus, sont pris en compte :

- la longueur de l'échangeur "length" (pour le calcul du nombre d'ébullition Bo dans la corrélation relative à l'évaporation) ;
- le facteur de surface ;
- l'efficacité d'ailette.

Trois options sont possibles pour le choix de la recherche du couple (Ac, dh) optimal : "Hot fluid" pour le seul fluide chaud, "Cold fluid" pour le seul fluide froid, et "Both Fluids" pour les deux à la fois. Dans ce dernier cas, des valeurs identiques des paramètres (Ac, dh, L, f et η_f) sont imposées aux deux fluides de l'échangeur (hypothèse raisonnable pour les échangeurs à plaques, a priori symétriques). Dans les autres cas, les valeurs des paramètres apparaissant sous "Other fluid settings" sont imposées, constantes, pour le fluide non sélectionné. Il est difficile de définir un indicateur de précision qui donne pleinement satisfaction, car les écarts entre les prédictions du modèle et les données expérimentales peuvent être estimés de nombreuses manières, sans qu'on puisse a priori savoir laquelle est la meilleure. Dans un échangeur, il semblerait naturel de s'intéresser aux écarts obtenus sur les températures de sortie des fluides, mais d'une part leur calcul en non-nominal peut se révéler relativement long, et d'autre part la pondération entre elles deux influencerait sur le critère. Pour accélérer le calcul, il nous a paru préférable de comparer les valeurs des surfaces d'échange prédites par le modèle en mode dimensionnement.

Le bouton "Run" lance donc les itérations au cours desquelles la surface de l'échangeur est déterminée en mode dimensionnement technologique, pour l'ensemble des fichiers de projet placés dans le répertoire "param_search". Les résultats de simulation sont stockés dans le fichier "simul.txt", qui peut être ouvert dans une feuille de calcul. Pour que le lancement des itérations soit possible, il faut que le fichier "simul.txt" soit fermé.

La ligne "Simuler et sauvegarder" du menu "Spécial" permet d'activer ou de désactiver la demande par ThermoOptim de la sauvegarde des projets simulés. En règle générale, il est préférable de désactiver cette fonction, pour éviter que cette requête soit effectuée à chaque chargement de projet, d'autant plus que le

paramétrage final correspond aux valeurs supérieures des intervalles de variation de Ac et dh . Une fois l'optimum trouvé, on peut relancer la procédure avec un nombre d'incrément égal à 1 pour les deux valeurs, afin de forcer la mise à jour des paramètres obtenus. Il est alors possible, en cliquant sur "Run", de sauvegarder les différents fichiers du répertoire "param_search", ce qui permet de les mettre à jour avec les valeurs trouvées.

Notons qu'il est aussi possible d'imposer ces valeurs en utilisant le mécanisme de modification de fichiers de projet présenté section 5.2.1.

Le menu File propose deux lignes qui permettent de sauvegarder le jeu complet des valeurs de l'écran de la figure 5.3.1 ou de le recharger.

5.3.3 Exploitation du fichier simul.txt

Le fichier "simul.txt" peut être copié dans une feuille de calcul du type de celle de la figure 5.3.2.

Les lignes du bas de la feuille de calcul permettent de calculer la moyenne, l'écart type et un indicateur de précision de l'identification, que nous appelons la dispersion sur la surface d'échange, égal au rapport de l'écart type à la moyenne. Dans l'exemple donné, le minimum de dispersion est de 3,16 %. Les premières lignes du fichier donnent les valeurs choisies pour le paramétrage des corrélations.

Pour rechercher le paramétrage le plus vraisemblable, il faut en essayer plusieurs et les comparer. L'expérience montre que la meilleure précision peut être obtenue avec de nombreuses valeurs de la surface (dans l'exemple ci-dessus, 6,15 m² et 7,51 m² sont équivalents). Une indication de la valeur nominale de la surface est donc nécessaire pour lever le doute.

5.3.4 Méthode d'identification

Pour réaliser une bonne convergence du processus d'identification, on procède de la manière suivante :

- on commence par choisir pour chaque fluide la configuration d'écoulement la plus pertinente, et on estime les paramètres Ac , dh , f , $length$... à partir des données géométriques, qui servent à paramétrer le fichier de projet initial ;
- on génère les fichiers de projet correspondant aux données expérimentales en utilisant les écrans présentés section 5.1 ;
- on teste un premier jeu de paramètres de l'échangeur avec tous les points disponibles, et on étudie les valeurs extrêmes des surfaces calculées ;
- sur la base de cet examen, on modifie le paramétrage pour réduire les écarts (sur un sous-ensemble représentatif des données s'il y en a beaucoup), et on optimise les paramètres, en réduisant la dispersion sans se soucier de la valeur de la surface moyenne ;
- on teste le jeu de paramètres obtenu sur l'ensemble des points ;
- on cherche enfin à obtenir la bonne valeur de la surface d'échange (correspondant à celle donnée par le constructeur) en s'inspirant de la démarche exposée section 5.3.5.

	1	2	3	4	5	6	7	8	9	10	11	12
1	arameter corr C1	a_re	b_Pr	c_visc	Hx length	transfer vaporization						
2		0.2	0.674	0.33	0.14	1	false					
3	Ac values	0.007	0.007	0.007	0.007	0.00910083	0.00910083	0.00910083	0.01183216	0.01183216	0.01183216	0.01183216
4	dh values	0.002	0.0029907	0.00447214	0.0066874	0.00447214	0.0066874	0.01	0.002	0.0029907	0.00447214	0.0066874
5	DX7503B.PR.	4.24388094	4.59427411	4.96557043	5.36341375	6.05113659	6.53147921	7.05451849	6.31936323	6.83308833	7.37590016	7.9559945
6	DX5603.PR.	4.19634637	4.57189145	4.97505775	5.41209285	6.04402538	6.57007919	7.14805922	6.21188332	6.75897222	7.34467162	7.9780039
7	DX15703B.PF.	4.19390421	4.54458387	4.91712218	5.31708964	5.99145728	6.474408	7.00082831	6.24346048	6.75785464	7.30238288	7.8857099
8	DX15703A.PF.	4.53438544	4.90196494	5.29127914	5.70838263	6.45623987	6.96080474	7.51018357	6.76761595	7.30836905	7.87958545	8.4901128
9	DX15603A.PF.	4.23222316	4.5701036	4.93165204	5.32022943	6.07009579	6.54752971	7.06377244	6.41513408	6.925738	7.4718865	8.0585333
10	DX18603.PR.	4.46439528	4.85218193	5.26675191	5.71377763	6.41436383	6.95421334	7.54371023	6.63940863	7.20787372	7.81405911	8.4661713
11	DX9703A.PR.	4.26589788	4.60283692	4.95618257	5.33130039	6.04360385	6.49647896	6.98636427	6.36041853	6.85473787	7.37143935	7.9183268
12												
13		4.30443332	4.66254812	5.04337372	5.45232376	6.15298894	6.64785616	7.18677665	6.42246773	6.94949055	7.50855644	8.1075510
14		0.13707124	0.14842975	0.16232622	0.17973918	0.19470224	0.21387267	0.23834737	0.20700904	0.22050912	0.2373836	0.2588870
15		3.18%	3.18%	3.18%	3.22%	3.30%	3.16%	3.22%	3.32%	3.22%	3.17%	3.16%
16		3.16%										3.19





Figure 5.3.2 : Résultats de simulation

5.5 References

Bernier Michel A. and Bourret Bernard, "Pumping Energy and Variable frequency Drives", ASHRAE Journal, December 1999.

6 EDITEUR DE STRUCTURES PRODUCTIVES

Nous avons implémenté dans Thermoptim un nouvel éditeur permettant de concevoir des structures et de les relier au simulateur et à l'éditeur de schémas pour pouvoir calculer proprement les allocations exergetiques de chaque composant et représenter les flux d'exergie échangés (figure 6.1).

Il est accessible à partir du menu "Spécial" de l'écran du simulateur (Ctrl B). On reconnaît sur sa palette l'icône des transfos-points  pour représenter les flux d'entrée et de sortie, celle des unités productives ou dissipatives , et celles des jonctions  et des embranchements .

Cet éditeur de structures productives étant analogue à l'éditeur de schémas, le détail de ses fonctionnalités élémentaires est donné dans le tome 1 du manuel de référence de Thermoptim.

On construit une structure productive soit à la main en sélectionnant les icônes sur la palette, en les plaçant sur le plan de travail, et en les connectant par des liens, de manière analogue à celle utilisée pour constituer un schéma, soit de manière automatique en générant une structure productive à partir du simulateur et du schéma physique.

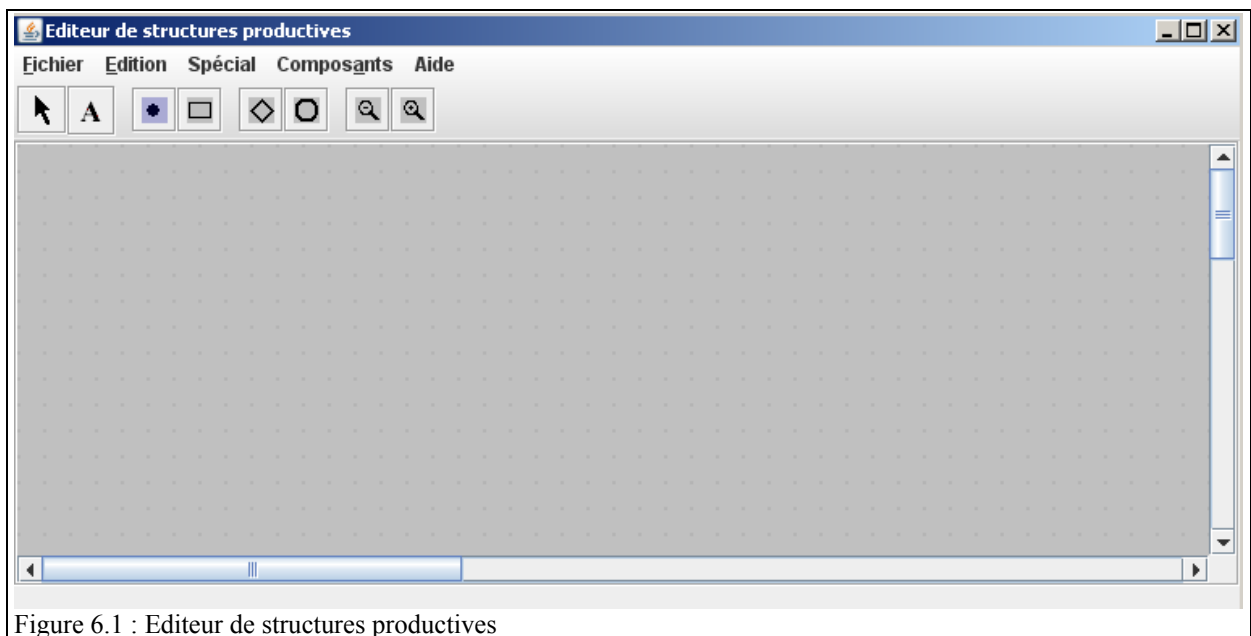


Figure 6.1 : Editeur de structures productives

6.1 Unités productives ou dissipatives

Les ressources exergetiques étant de trois types possibles, et les produits de deux, une unité productive ou dissipative (UPD) est représentée par un rectangle muni de trois ports d'entrée et deux de sortie (figure 6.2), dont les couleurs sont différenciées pour distinguer les différents types de flux exergetiques et permettre de distinguer graphiquement les ressources et produits.

L'affectation des couleurs est la suivante pour les ports d'entrée : rouge pour les combustibles, bleu pour la puissance mécanique, et vert pour l'exergie d'un fluide ; pour les ports de sortie : bleu pour la puissance mécanique, et vert pour l'exergie d'un fluide.

Les valeurs des flux exergetiques peuvent apparaître sur les liens, qui sont colorés selon le code de couleurs retenu pour les ports d'entrée et de sortie.

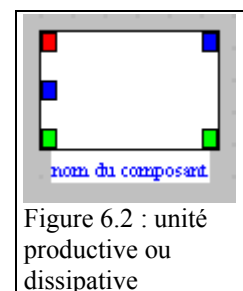


Figure 6.2 : unité productive ou dissipative

6.2 Jonctions et embranchements

L'exergie du fluide qui traverse un composant peut augmenter ou diminuer. Nous l'appellerons **extenseur d'exergie** si elle augmente, et **réducteur d'exergie** si elle diminue.

Un **extenseur** est représenté par une unité productive avec une jonction en sortie, à laquelle arrivent deux liens : celui issu du port vert de l'unité productive (flux exergétique de sortie), et celui correspondant au flux exergétique d'entrée (figure 6.3).

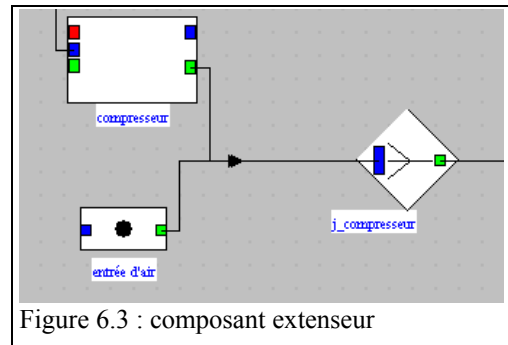


Figure 6.3 : composant extenseur

Sur cette figure, l'exergie totale disponible en sortie du compresseur est la somme de deux valeurs : d'une part le flux exergétique en entrée du composant, représenté par la transfo-point "entrée d'air", et d'autre part la variation d'exergie communiquée au fluide dans le compresseur, lequel la reçoit sous forme mécanique. Le pseudo-mélangeur ou jonction "j_compresseur" sert à modéliser cette sommation.

Un **réducteur** est représenté par une unité productive avec un embranchement en entrée, dont partent deux liens : celui aboutissant au port vert de l'unité productive ou dissipative (flux exergétique d'entrée), et celui correspondant au flux exergétique de sortie (figure 6.4).

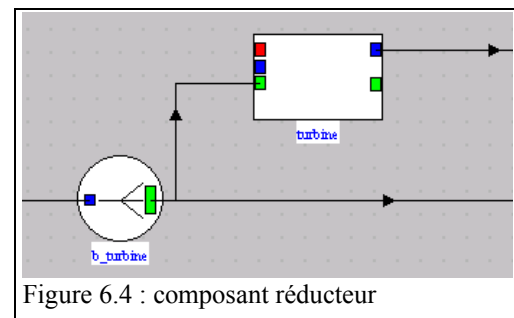


Figure 6.4 : composant réducteur

Sur cette figure, l'exergie totale disponible en amont de la turbine se décompose en deux parties : d'une part celle qui sert de "combustible" à la turbine et est convertie sous forme mécanique, et d'autre part celle qui reste disponible en sortie de turbine. Le pseudo-diviseur ou embranchement "b_turbine" sert à modéliser cette répartition.

On notera que, pour les composants extenseurs (ceux pour lesquels $\Delta x_h > 0$), $P_i = \Delta x_h$. Pour la turbine (pour laquelle $\Delta x_h < 0$) $P_i = \tau$. Les laminages étant purement dissipatifs, $P_i = 0$.

6.3 Automatisation de la création de la structure productive

Il est possible de construire à la main avec Thermoptim la structure productive d'un système énergétique, mais ce n'est pas la meilleure manière d'opérer si on en connaît le schéma physique, comme c'est généralement le cas.

La structure productive peut en effet très largement se déduire du schéma physique (schéma Thermoptim) et du paramétrage thermodynamique (projet Thermoptim). A la condition que la température T_0 de l'environnement soit correctement initialisée (menu Aide / paramètres globaux du simulateur), ce dernier sert notamment à distinguer les composants extenseurs et réducteurs d'exergie ainsi qu'à calculer les bilans exergétiques des UPD.

Dans une première étape, en activant la ligne « Transfert du schéma » du menu « Spécial », on crée toutes les unités productives en respectant les règles de correspondance énoncées ci-dessus. Il faut bien entendu qu'un projet Thermoptim et son schéma aient été préalablement ouverts et recalculés.

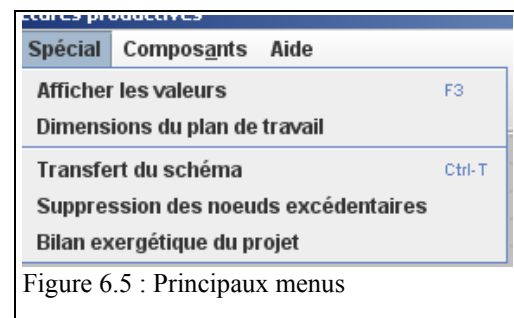


Figure 6.5 : Principaux menus

Notons qu'en créant la structure productive, il est fréquent que l'on introduise des jonctions en série et des embranchements en série, qui peuvent très bien être regroupés pour simplifier la structure productive. Il suffit pour cela que plusieurs extenseurs ou plusieurs réducteurs soient disposés en série dans le schéma. C'est par exemple le cas lorsqu'un échangeur à changement de phase comme un générateur de vapeur est représenté par trois échangeurs en série (économiseur, vaporiseur, surchauffeur).

Aussi est-il possible, dans une deuxième étape, de sélectionner dans l'éditeur de structures les pseudos-nœuds en série, et de les fusionner en activant la ligne « Suppression des nœuds excédentaires » du menu « Spécial ». Afin d'éviter des erreurs de représentation, vous pouvez choisir de ne pas automatiquement supprimer les véritables nœuds provenant du schéma physique, sachant qu'ils peuvent l'être à la main dans un second temps.

Dans une troisième étape, on finit de renseigner la structure productive, en paramétrant les écrans des bilans exergétiques présentés ci-dessous. Par exemple, les entrées d'exergie bleues (puissance mécanique) ne sont pas connectées à ce stade, tout comme ne sont pas définies les températures des sources externes avec lesquelles des exergies-chaleur sont échangées.

6.4 Bilans exergétiques de composants

Dans Thermoptim, le lien logique entre la structure productive et le simulateur se fait par le nom du composant. Il faut donc nommer l'UPD du même nom que l'élément correspondant du simulateur.

A chaque unité productive est associé un écran de bilan exergétique de composant, qui permet de préciser certains paramétrages en vue de calculer les ressources et les produits exergétiques, ainsi que les irréversibilités et le rendement exergétique du composant (figures 6.6 et 6.7).

Un double-clic sur l'unité productive ouvre cet écran qui permet de préciser certains paramétrages en vue de calculer les ressources et les produits exergétiques, ainsi que les irréversibilités et le rendement exergétique du composant, ce que, par raccourci, nous désignerons par dresser le bilan exergétique du composant.

Field	Value
Resource ΔXh^+	1 556,221
Product τ	1 326,01
Exergy efficiency	0,852071
Irreversibilities	230,211

Figure 6.6 : Bilan exergétique d'un composant turbine

La figure 6.6 montre l'écran d'un composant turbine, dont le bilan exergétique peut être déterminé sans paramétrage spécifique : la ressource exergétique provient de la variation d'exergie du fluide qui le traverse (ΔXh^+), le produit est la puissance mécanique (τ), le rendement exergétique et la valeur des irréversibilités s'en déduisent directement.

Lorsque le composant est du type "échange", différentes possibilités existent.

Lorsqu'il s'agit d'une **entrée ou une sortie de fluide**, ce qui correspond à une transfo-point reliée uniquement en amont ou en aval, son exergie totale (physique et chimique) est prise en compte comme indiqué en annexe 2. Notez bien que cela peut jouer sur l'équilibrage des bilans globaux car les modes de calcul des diverses propriétés exergétiques ne sont pas les mêmes.

S'il s'agit d'une **sortie de fluide** avec rejet dans l'environnement, l'exergie est perdue, ce qui constitue une irréversibilité. Si au contraire cette exergie est valorisée en externe, il s'agit d'un produit exergétique sans irréversibilité. L'écran de calcul des exergies est muni d'une option permettant de spécifier cette situation (Valuable exergy).

Lorsqu'il y a des **échanges de chaleur internes**, comme c'est le cas dans les cycles combinés, le calcul de la perte exergétique demande aussi à être modifié : elle est égale, en valeur absolue, à la somme des variations d'exergie des deux fluides qui échangent de la chaleur. L'écran des bilans exergétiques comporte une option (Internal exchange) pour spécifier ce paramétrage, qui est sélectionné par défaut lorsqu'un échangeur existe dans le schéma physique (figure 6.7).

Enfin, lorsque l'échange de chaleur a lieu avec une **source d'énergie externe**, l'exergie-chaleur mise en jeu doit être évaluée, ce qui suppose que la température de la source soit connue. L'écran de calcul des exergies est muni d'une option permettant de spécifier cette situation (External source), et d'un champ destiné à entrer la température de la source.

Pour les **laminages**, l'exergie reçue est l'exergie totale entrante, et l'exergie fournie l'exergie totale sortante. L'irréversibilité est égale à leur différence.

Etant donné que les fluides extenseurs d'exergie sont représentés par une UPD couplée à un mélangeur, il faut bien distinguer ces pseudos-mélangeurs des mélangeurs réels du système physique, l'irréversibilité qui prend place dans un vrai mélangeur pouvant provenir de différentes sources :

- diminution de la pression d'un fluide (perte de charge)
- mélange avec changement de composition

On trouvera en annexe 2 un tableau montrant comment les différents types de composants du noyau de Thermoptim sont représentés sous forme d'UPD.

Les **composants externes** peuvent aussi être représentés dans une structure productive. On se référera au manuel de référence tome 3 pour plus de précisions à ce sujet.

6.5 Génération des bilans exergétiques

Une fois une structure productive établie, on peut soit calculer un par un les **bilans exergétiques** des diverses UPD, soit ouvrir un écran permettant de le faire globalement pour l'ensemble du projet en activant la ligne « Bilan exergétique du projet » du menu « Spécial ». Thermoptim construit alors un tableau dont les lignes correspondent à chacun des composants, plus une pour le système dans son ensemble. Cette dernière présente comme ressource la somme des exergies chaleurs positives et du travail utile fournis $\Sigma (\Delta x_{qj}^+) + \Sigma (\tau_j^+)$, comme produit la somme des exergies sortant du système, le rendement exergétique du système, et la somme des irréversibilités. Si les règles d'établissement du bilan exergétique retenues dans Thermoptim ne lui donnent pas satisfaction, l'utilisateur peut l'exporter et le retravailler dans un tableur.

Dans le bilan exergétique du projet, les variations d'exergie des fluides couplés par un échangeur sont indiquées entourées de parenthèses, les deux transfos d'un échangeur étant regroupées l'une à côté de l'autre pour plus de clarté (figure 6.8).

Des exemples de structures productives et les bilans exergétiques correspondants pour une centrale à vapeur, une machine de réfrigération et une turbine à gaz avec cogénération sont donnés dans le chapitre 10 (partie 3) du livre Systèmes Energétiques

Figure 6.7 : Bilan exergétique d'une transfo échange avec échange interne

composant	Resource	Product	exergy efficiency	irreversibilities	% total	settings
combustible	897,041	0	0	0	0	
compresseur	442,71	402,238	0,908581	40,472	0,08269	
turbine	796,411	760,94	0,955462	35,471	0,07247	
chambre de combustion	0	613,557	0,683978	275,983	0,5639	
eau froide	4,593	0	0	0	0	
vapeur chaude	114,658	114,658	0	0	0	
HRSG gaz	157,839	0	0	(157,839)	(0,3225)	HRSG
HRSG eau	0	110,065	0	(-110,065)	(-0,2249)	HRSG
sortie gaz	0	0	0	89,751	0,1834	loss
global	901,634	432,888	0,4572	489,451	1	

Figure 6.8 : Bilan exergétique d'une turbine à gaz avec cogénération

ANNEXE 1 : Classes ExtPilot, PointThopt, CorpsThopt

Dans cette annexe, nous donnons quelques indications sur les trois classes ExtPilot, PointThopt et CorpsThopt qui ont été définies pour faciliter la création des classes externes avancées, et notamment des pilotes.

Classe ExtPilot

C'est la classe mère de tous les pilotes. Afin de sécuriser le paramétrage des différentes méthodes de mise à jour des objets du noyau de Thermoptim qu'elle propose, elle définit un certain nombre de booléens sous forme directement compréhensible :

```
boolean RECALCULATE=true, UPDATE_T=true, UPDATE_P=true, UPDATE_X=true,
UPDATE_ETA=true, UPDATE_UA=true, UPDATE_EPSI=true, UPDATE_FLOW=true,
IS_SET_FLOW=true, UPDATE_DTMIN=true;
```

De cette manière, le code suivant d'une mise à jour d'un point :

```
avalFroid.update(UPDATE_T,!UPDATE_P,!UPDATE_X);
```

se lit mettre à jour la seule température du point avalFroid, ce qui est beaucoup plus lisible que :

```
avalFroid.update(true, false, false);
```

```
public void updateprocess(String name, String type, boolean recalculate, boolean isSetFlow, boolean
updateFlow, double flow,
    boolean updateParam1, double param1, boolean updateParam2, double param2)
```

Disponible avec plusieurs signatures, cette méthode met à jour et recalcule une transfo. Si recalculate vaut true, elle est recalculée, si isSetFlow vaut true, son débit est imposé, si updateFlow vaut true, son débit est modifié et vaut flow, si updateParam vaut true, la valeur param est imposée. Le sens de param dépend du type et peut être retrouvé en se référant à la méthode updateProcess() de Projet, telle que définie dans le tome 3 manuel de référence.

Par exemple, la ligne suivante met à jour la transfo de nom compressorName, de type Compression, demande le recalcul, impose le débit en modifiant sa valeur, et modifie le rendement isentropique :

```
updateprocess(compressorName, "Compression",RECALCULATE,IS_SET_FLOW, UPDATE_FLOW,
massFlow, UPDATE_ETA, eta_is);
```

De manière analogue, existent des méthodes de mise à jour et recalcul pour les échangeurs et les nœuds :

```
public void updateHx(String name, boolean recalculate, boolean updateUA, double UA,
    boolean updateEpsi, double epsi, boolean updateDTmin, double DTmin)
```

```
public void updateNode(String name, boolean recalculate, boolean updateEfficiency, double epsi)
```

Les deux méthodes transferTechnoData() et setupTechnoDesigns() permettent d'initialiser le pilote et les écrans technologiques lors de l'ouverture d'un fichier de projet existant.

La première méthode regroupe toutes les lignes du fichier de projet relatives aux écrans technologiques et les charge dans le pilote en attendant qu'il soit construit, et la seconde décode ces lignes pour effectuer ses initialisations, en passant le relais aux méthodes standard des TechnoExch, qui font quant à elles appel aux FlowConfig.

La dernière ligne de setupTechnoDesigns(), proj.bindTechnoDesigns(vSettings), sert à relier les TechnoDesign utilisés par le pilote avec les composants du noyau, afin qu'ils puissent être affichés à partir de leurs écrans et des tables générales de l'écran d'affichage technologique et de simulation.

```

void setupTechnoDesigns(Vector vTechnoDesign){
    Vector vSettings=new Vector();
    if(technoDesignData.equals(""))setupTechnoDesigns();
    else{//cas où il faut relire les données
        StringTokenizer st = new StringTokenizer(technoDesignData,"\n");
        while(st.hasMoreTokens()){
            String ligne=st.nextToken();
            String type=Util.extr_value(ligne,"compType");
            String comp=Util.extr_value(ligne,"component");
            String value="";
            if(type.equals("HeatEx")){//échangeurs : 3 lignes dans ce cas
                value=Util.extr_value(ligne,"&Design_value");
                for(int j=0;j<vTechnoDesign.size();j++){
                    TechnoHx etd;
                    Object[] obj1=new Object[5];
                    obj1=(Object[])vTechnoDesign.elementAt(j);
                    etd=(TechnoHx)obj1[1];
                    String etdType=(String)obj1[4];
                    String etdComp=(String)obj1[0];
                    if((comp.equals(etdComp)) && (type.equals(etdType))){
                        if(!(value==null)){
                            etd.&Design_value.setText(value);
                        }
                        ligne=st.nextToken();
                        etd.techc.readCompParameters(ligne);
                        ligne=st.nextToken();
                        etd.techf.readCompParameters(ligne);
                        Object[] obj=new Object[3];
                        obj[0]=comp;
                        obj[1]=type;
                        obj[2]=etd;
                        vSettings.addElement(obj);
                        break;
                    }
                }
            }
        }
    }
    else{//autres TechnoDesign
        for(int j=0;j<vTechnoDesign.size();j++){
            ExtTechnoDesign etd;
            Object[] obj1=new Object[5];
            obj1=(Object[])vTechnoDesign.elementAt(j);
            etd=(ExtTechnoDesign)obj1[1];
            String etdType=(String)obj1[4];
            String etdComp=(String)obj1[0];
            if((comp.equals(etdComp)) && (type.equals(etdType))){
                etd.readCompParameters(ligne);
                Object[] obj=new Object[3];
                obj[0]=comp;
                obj[1]=type;
                obj[2]=etd;
                vSettings.addElement(obj);
                break;
            }
        }
    }
}
proj.bindTechnoDesigns(vSettings);

```

Classe *PointThopt*

Cette classe permet de créer dans une classe externe des sortes de clones des points du noyau de Thermoptim, afin d'avoir un accès aisé à leurs valeurs, qui ne sont pas directement accessibles. Elle apporte davantage de confort et de lisibilité que ne le fait l'utilisation des méthodes `getProperties()` et `updatePoint()` de `Projet`, documentées dans le tome 3 manuel de référence. Elle possède des double dans lesquels sont stockés les équivalents des variables d'état du noyau :

```
double W,Epsi,QPrime,Tprime,Tr,VPrime,Cond, M_sec, corrFactor;
CorpsThopt corps;
```

Lors de la construction, on spécifie la référence au projet et le nom du point dans le simulateur, qui permet à Thermoptim de savoir quel point est concerné. Attention, s'il y a une erreur à ce niveau, le `PointThopt` ne peut être correctement instancié.

```
public PointThopt(Projet proj, String name){
    this.proj=proj;
    this.name=name;
    try{
        getProperties();
    }
    catch(Exception e){
        String msg = "Error constructing the PointCorps name: "+ name;
        JOptionPane.showMessageDialog(new JFrame(), msg);
        e.printStackTrace();
    }
    corps=new CorpsThopt(proj, nomCorps,lecorps);
}
```

`PointThopt` propose lui aussi des méthodes de mise à jour de son équivalent du simulateur, fonctionnant sur le même principe que celles de `ExtPilot`, avec des signatures variables selon le nombre de valeurs à modifier.

```
public void update(boolean updateT, boolean updateP, boolean updateX, boolean updateCorrFactor, boolean
melHum, String task, double value)
```

Une version allégée la plus courante a servi d'exemple à l'utilisation des booléens explicites d'`ExtPilot` :

```
public void update(boolean updateT, boolean updateP, boolean updateX){
    update(updateT, updateP, updateX, false, false, "", 0.);
}
```

Classe *CorpsThopt*

Cette classe permet de créer dans une classe externe des sortes de clones des corps du noyau de Thermoptim, afin d'avoir un accès aisé à leurs valeurs, qui ne sont pas directement accessibles. Elle apporte davantage de confort et de lisibilité que ne le fait l'utilisation des méthodes `getProperties()` et `updateSubstance()` de `Projet`, documentées dans le tome 3 manuel de référence. Elle possède des double dans lesquels sont stockés les équivalents des variables d'état du noyau :

```
public double T,P,X,V,U,H,S,M,M_sec,TC,PC,VC;
```

Lors de la construction, on spécifie la référence au projet et le nom du corps dans le simulateur, qui permet à Thermoptim de savoir quel corps est concerné. Attention, s'il y a une erreur à ce niveau, le `CorpsThopt` ne peut être correctement instancié.

```
public CorpsThopt(Projet proj, String name, Corps lecorps){
    this.proj=proj;
    this.name=name;
    this.lecorps=lecorps;
```

```
}
```

Un exemple d'instanciation est donné dans le constructeur de PointThopt.

La méthode **getSubstProperties()** permet de mettre à jour les valeurs après un calcul effectué sur la variable lecorps. L'exemple ci-dessous montre l'utilisation d'un CorpsThopt faisant appel au PointThopt "amont" :

```
CorpsThopt corps=new CorpsThopt(proj,amont.nomCorps, amont.lecorps);
corps.lecorps.CalcPropCorps(amont.T,amont.P,1.);
corps.getSubstProperties();
```

Les fonctions usuelles sont alors directement accessibles : par exemple, l'enthalpie vaut corps.H.

CLASSES DE DIMENSIONNEMENT TECHNOLOGIQUE

Les classes relatives aux compresseurs volumétriques et aux échangeurs ayant été présentées section 3.2, nous ne montrerons ici que les autres.

Rappelons que ces classes ne doivent être considérées que comme des ébauches destinées à être affinées par les utilisateurs souhaitant étudier le dimensionnement technologique et le comportement des systèmes en régime non-nominal, et non pas comme des modèles parfaitement validés. Notre objectif est simplement de faciliter leur démarche en leur fournissant de quoi démarrer.

Pour introduire de nouvelles classes, il suffit soit de sous-classer celles qui existent, soit d'en créer sur le même modèle et de les placer dans extThopt.zip ou extUser.zip pour la version 1.7 et extThopt2.zip ou extUser2.zip pour la version 2.7.

Classes représentatives des turbines

La modélisation en non-nominal des turbines est un problème qui peut être abordé à des niveaux de difficulté variables. C'est pourquoi nous avons implémenté une série de modèles différents, permettant de graduer la finesse d'approche en fonction d'une part des données disponibles et d'autre part du niveau de l'utilisateur :

- la classe de base s'appelle TechnoSimpleTurb. Elle est basée sur la règle du cône et ne s'applique en toute rigueur qu'à une turbine mono-étagée ;
- la classe TechnoMultiStageTurb étend la précédente en considérant que la règle du cône de la turbine multi-étagée est la même que celle d'une turbine mono-étagée, le rapport de détente à prendre en compte étant la racine nième du rapport de détente total, n étant le nombre d'étages. Si n=1, le comportement est le même que la classe précédente ;
- la classe TechnoTurb étend la précédente, en permettant de prendre en compte d'une part l'existence de pertes par vitesse résiduelle (PVR) en sortie de turbine, et d'autre part une détérioration du rendement isentropique en zone humide, selon la règle de Baumann ;
- la classe MultiStageMappedTurbine étend la précédente, la règle du cône étant cette fois-ci remplacée par une cartographie représentée par un ajustement numérique à une vingtaine de paramètres.

Figure A.1 : Ecran de dimensionnement de base d'une turbine monoétagée

TechnoSimpleTurb

Cette classe définit les paramètres de dimensionnement technologique pour les turbines (figureA.1).

$$\frac{\dot{m} \sqrt{T_{in}}}{P_{in}} = Cste \quad (A.1)$$

$$T_{in} \frac{\dot{m}^2}{K} = P_{in}^2 - P_{out}^2 \quad (A.2)$$

La signification des champs est la suivante :

- Stodola constant est la valeur du coefficient de Stodola. Si l'option "choked turbine" est cochée, le débit est considéré comme choqué, et la formule retenue est (A.1). Sinon, c'est (A.2) ;
- lorsque η est représenté par une équation à 3 paramètres, η_{\max} est la valeur maximale, η_{\lim} la limite pour les taux de détente très élevés, et τ_{\max} l'abscisse du maximum. Si les deux premières valeurs sont égales, le rendement isentropique est constant ;
- rotation speed est la vitesse de rotation ;

TechnoMultiStageTurb

Son écran se distingue du précédent par l'apparition d'un nouveau champ, représentant le nombre d'étages de la turbine (figure A2).

La formule de Stodola prise en compte est plus générale que (A2), et prend en compte le coefficient polytropique de la détente.

Figure A.2 : Ecran de dimensionnement de base d'une turbine multiétagée

TechnoTurb

Son écran (figure A3) se distingue du précédent par l'apparition de plusieurs nouveaux champs, permettant comme indiqué plus haut de prendre en compte d'une part l'existence de pertes par vitesse résiduelle (PVR) en sortie de turbine, et d'autre part une détérioration du rendement isentropique en zone humide, selon la règle de Baumann (A3).

Les valeurs des pertes par vitesse résiduelle apparaissent sur la gauche de l'écran.

Figure A.3 : Ecran de dimensionnement d'une turbine multiétagée

$$\eta_{\text{hum}} = \eta_{\text{sec}} (1 - \alpha (1 - x)) \quad (\text{A.3})$$

La signification des champs est la suivante :

- alpha Baumann est la valeur du coefficient α de l'équation (A.3) ;

- pour le calcul des pertes par vitesse résiduelle, il faut de plus fournir la section de sortie et le diamètre de la turbine, ainsi que l'angle β de sortie.

Classe MultiStageMappedTurbine

Cette classe, qui hérite de TechnoTurb, définit les paramètres de dimensionnement technologique pour les turbines représentées par une cartographie détaillée, telle que définie dans une note de modélisation⁴.

L'écran technologique (figure A.4) possède dans sa partie supérieure droite un libellé ("Turbine data") qui donne accès par double-clic à l'écran permettant d'afficher la série de coefficients décrivant les caractéristiques de débit et de rendement isentropique (classe TurbineMapDataFrame, figure A.5).

Figure A.4 : Ecran de dimensionnement d'une turbine avec cartographie

Figure A.5 : Ecran de définition des paramètres des cartographies de turbines

⁴ <http://www.thermoptim.org/SE/seances/C01/ModelisationSimplifieeTurbomachines.pdf>

A droite de la figure A.4, apparaît en médaillon une image correspondant aux caractéristiques retenues, si elle a été chargée, et sinon le libellé "Turbine maps" qui permet par double-clic d'ouvrir un écran de chargement d'image (classe MapImage).

Les fichiers de données et d'images des cartographies doivent être placés dans le répertoire "maps" à partir duquel travaillent les deux classes. Leurs noms doivent être sauvegardés par le pilote.

La cartographie utilisée est ici exprimée par rapport au débit normé. Lors du dimensionnement, on cale le point nominal de la cartographie sur celui de la transfo détente (dans un deuxième temps, on pourra en déduire les dimensions, mais ce n'est pas encore effectué), ce qui permet d'obtenir les valeurs "eta design", "Rp design" et "m design" affichées à gauche de l'écran.

Au dessous d'elles apparaissent les valeurs adimensionnelles permettant de situer le point de fonctionnement sur la cartographie utilisée. Leurs valeurs sont bien évidemment corrigées en fonction des conditions d'admission dans la turbine.

La valeur de la vitesse de rotation peut être modifiée depuis l'écran du pilote si on le souhaite.

Classe TechnoNozzle

Cette classe définit les paramètres de dimensionnement technologique pour les tuyères (figure A.6).

En régime non-nominal, une tuyère peut être modélisée de manière analogue à une turbine, et caractérisée par une constante de Stodola et un

rendement isentropique. Son écran correspond donc à la partie supérieure de celui d'une turbine mono-étagée.

Figure A.6 : Ecran de dimensionnement d'une tuyère

Classe TechnoTurboCompr

Cette classe définit les paramètres de dimensionnement technologique pour les turbocompresseurs.

L'écran technologique (figure A.7) fait appel à une série de coefficients décrivant les deux caractéristiques.

$$\text{débit corrigé réduit dans le compresseur} \quad mc = \frac{\dot{m} T_1}{K_c N P_1} \quad (\text{A.4})$$

$$\text{caract. réduite du compresseur} \quad \psi = \psi_{\text{coeff}} (mc - \phi \psi_{\text{max}})^2 + \psi_{\text{max}} \quad (\text{A.5})$$

Les paramètres à définir sont les suivants :

- K_c , caractéristique du compresseur (équation (A.4)) ;
- le diamètre du compresseur ;
- ψ est représenté par une branche de parabole, définie (équation (A.5)) par ψ_{\max} , valeur maximale, ψ_{coeff} , et $\phi\psi_{\max}$, abscisse du maximum du débit corrigé m_c ;
- η est aussi représenté par une branche de parabole, définie par η_{\max} , valeur maximale, η_{coeff} , et $\phi\eta_{\max}$, abscisse du maximum du débit corrigé m_c .

Figure A.7 : Ecran de dimensionnement d'un turbocompresseur

Les valeurs du rendement isentropique, du débit réduit et du débit massique apparaissent sur la droite de l'écran.

Classe MultiStageMappedTurboCompr

Cette classe, qui hérite de TechnoTurboCompr, définit les paramètres de dimensionnement technologique pour les turbocompresseurs représentés par une cartographie détaillée, telle que définie dans la note de modélisation citée précédemment.

L'écran technologique (figure A.8) possède dans sa partie supérieure droite un libellé ("Compressor data") qui donne accès par double-clic à l'écran permettant d'afficher la série de coefficients décrivant les

Figure A.8 : Ecran de dimensionnement d'un turbocompresseur avec cartographie

caractéristiques de débit et de rendement isentropique (classe TurboComprMapDataFrame, figure A.9).

A droite de la figure A.9, apparaît en médaillon une image correspondant aux caractéristiques retenues, si elle a été chargée, et sinon le libellé "Compressor maps" qui permet par double-clic d'ouvrir un écran de chargement d'image (classe MapImage).

Les fichiers de données et d'images des cartographies doivent être placés dans le répertoire "maps" à partir duquel travaillent les deux classes. Leurs noms doivent être sauvegardés par le pilote.

La cartographie utilisée est ici exprimée par rapport au débit normé. Il faut entrer dans le fichier de données l'ensemble des valeurs de référence : P_0 , T_0 , N_{ref} , $R_{p,\text{ref}}$, m_{ref} , $\eta_{\text{is_ref}}$, ainsi que l'intervalle de variation de la vitesse réduite (ici 1,2 - 1,8).

Lors du dimensionnement, on cale le point nominal de la cartographie sur les conditions de la transfo compression (dans un deuxième temps, on pourra en déduire les dimensions, mais ce n'est pas encore effectué), ce qui permet d'obtenir la vitesse de rotation du compresseur et les valeurs "eta design", "Rp design" et "m design" affichées à gauche de l'écran, et éventuellement modifiables.

Au dessus d'elles apparaissent les valeurs adimensionnelles permettant de situer le point de fonctionnement sur la cartographie utilisée. Leurs valeurs sont bien évidemment corrigées en fonction des conditions d'admission dans le compresseur.

En dessous de ces valeurs apparaît le champ permettant de définir le nombre d'étages si on utilise une cartographie générique d'étage (ici 5).

La valeur de la vitesse de rotation du compresseur peut être modifiée depuis l'écran du pilote si on le souhaite.

Turbocompressor map data Nmin 1.2 Nmax 1.8 file: dataCentrifConan_1S_Isentr.txt

Surge line

y	x
alpha -0.136921682	alpha -2.514285714
beta 1.512965429	beta 3.714285714
gamma -0.21353161	gamma -0.952380952

Max flow line

y	x
alpha 0.76392513	alpha -3.644444444
beta 0.062459365	beta 5.904761905
gamma 0.320056518	gamma -1.746031746

Compression ratio map

a	b	c	d
alpha -1.08253966	alpha 2.110757872	alpha 3.060368542	alpha 0.88229885
beta 1.016891184	beta -2.885820961	beta 2.91882978	beta -0.541643701
gamma -1.191929584	gamma 0.737635233	gamma -0.668278807	gamma -0.37140948

Isentropic efficiency map

a1	a2	a3	a4
alpha 2.832173549	alpha 7.416629458	alpha -1.846398981	alpha 2.434642322
beta -5.514708413	beta -13.00965072	beta 12.2213757	beta -3.654899778
gamma 2.011299506	gamma 5.039113852	gamma -4.761350908	gamma 1.320385439

Reference settings

P0 (bar) 1.4	T0 (K) 260	Nref 20000
Rp ref 1	m ref 1.2	eta ref 1

Load settings Open new file Cancel

Figure A.9 : Ecran de définition des paramètres des cartographies de turbocompresseur

Remarques sur les valeurs de référence des fichiers de cartographie

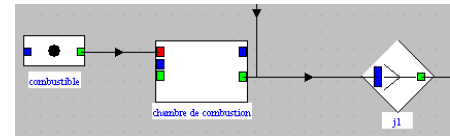
Les dernières lignes des fichiers de cartographie (bas d'écran des figures A.5 et A.9) fournissent les valeurs de référence qui permettent d'adapter une cartographie à un problème donné. Les pression et température d'entrée et la vitesse de rotation correspondent généralement à celles du simulateur pour le point nominal. Le rapport de compression/détente et le rendement isentropique/polytropique devraient être égaux à 1 si la cartographie correspond bien au compresseur utilisé, mais ils peuvent être modifiés si nécessaire. La valeur du débit de référence sera généralement proche de celle du débit du compresseur au point nominal, étant donné que les cartographies ont été établies pour des débits normés.

ANNEXE 2 REPRESENTATION DES COMPOSANTS SOUS FORME D'UNITES PRODUCTIVES

Composants extenseurs d'exergie

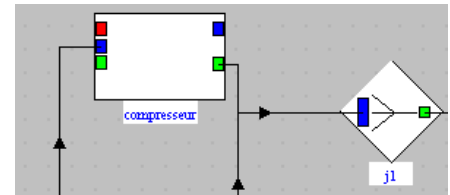
Composant F P

Combustion xq^+ Δxh



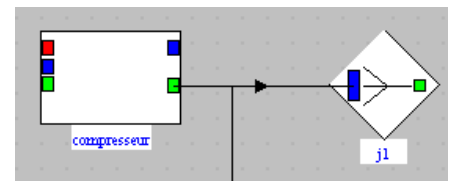
Une chambre de combustion convertit l'exergie du combustible xq^+ (représentée par un lien entrant dans le port rouge) en accroissement d'exergie Δxh du fluide qui la traverse (thermique moins pertes de charge). Le lien sortant provient donc du port vert. Il faut placer en aval une jonction dont l'autre entrée vient du comburant tandis que le combustible est connecté au port d'entrée rouge. L'exergie chimique éventuellement restante se retrouve dans les fumées et est ainsi prise en compte en aval.

Compresseur si τ interne τ^+ Δxh



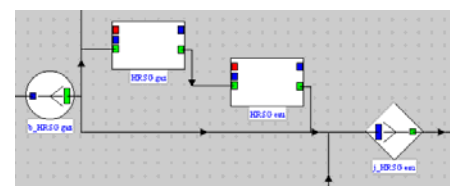
Un compresseur (ou une pompe) convertit la puissance mécanique τ^+ qu'il reçoit en accroissement d'exergie Δxh du fluide qui le traverse. Il faut donc placer en aval une jonction dont les autres entrées sont par exemple le comburant et la chambre de combustion. Le lien sortant provient du port vert.

Compresseur si τ externe τ^+ Δxh



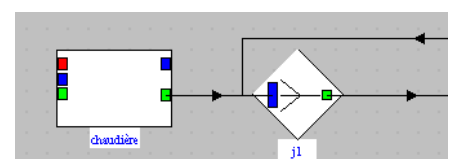
Si la puissance reçue est interne au système, elle est représentée par un lien entrant dans le port bleu. Si elle est externe, elle n'apparaît pas, mais est signalée dans l'écran de calcul exergétique.

Echangeur interne
fluide chaud en amont
fluide froid en aval Δxh^+ Δxh



Dans un échangeur modélisé en interne comme couplage entre deux transfos "échange", la ressource correspond à la variation d'exergie du fluide chaud, et le produit à celle du fluide froid. Il en résulte que le fluide chaud doit correspondre à l'une des branches d'un embranchement, et le fluide froid à l'une des branches d'une jonction. Les deux composants sont reliés comme indiqué sur la figure.

Echangeur externe $Q > 0$
 $T_k > T_0$ xq^+ Δxh



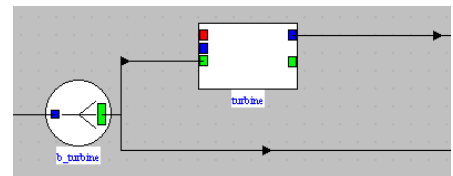
Dans un échange avec une source externe à température supérieure à celle de l'environnement, apportant de chaleur au système, la ressource est l'exergie-chaleur xq^+ échangée avec la source. Le produit est l'accroissement d'exergie Δxh du fluide qui traverse le composant. Il faut placer en aval une jonction. Le lien sortant provient du port vert.

Composants réducteurs d'exergie

Turbine

Δxh^+

τ

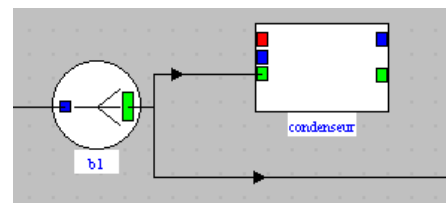


Une turbine convertit en puissance mécanique τ la diminution d'exergie Δxh^+ du fluide qui la traverse (représentée par un lien entrant dans le port vert). Il faut donc placer en amont un embranchement avec au moins une autre sortie, par exemple un rejet à l'atmosphère des gaz détendus, ou un condenseur s'il s'agit d'une vapeur. Le lien sortant représentant le travail provient du port bleu.

Echangeur externe $Q < 0$

$T_k > T_0$

Δxh^+

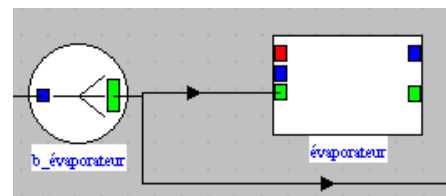


Dans un refroidissement avec une source externe à température supérieure à celle de l'environnement, la ressource est la diminution d'exergie Δxh^+ du fluide qui le traverse (représentée par un lien entrant dans le port vert), et le produit est l'exergie-chaleur cédée par le système à la source externe, dont la température de la source externe doit être indiquée dans l'écran de calcul exergetique. Il faut placer en amont un embranchement avec au moins une autre sortie.

Echangeur externe $Q > 0$

$T_k < T_0$

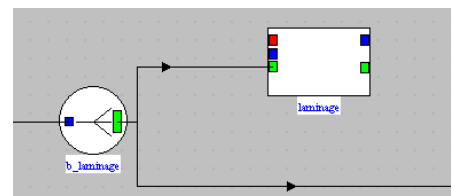
Δxh^+



Dans un échange avec une source externe externe à température inférieure à celle de l'environnement, avec apport de chaleur au système, la ressource est la diminution d'exergie Δxh^+ du fluide qui le traverse (représentée par un lien entrant dans le port vert), et le produit est l'exergie-chaleur cédée par le système à la source externe, dont la température de la source externe doit être indiquée dans l'écran de calcul exergetique. Il faut placer en amont un embranchement avec au moins une autre sortie.

Laminage

$\Delta xh^+ \quad 0$



Dans un laminage, l'exergie totale du fluide diminue entre l'entrée et la sortie du composant, du fait de la baisse de pression à enthalpie constante. Il faut placer en amont un embranchement. Il s'agit d'un composant dissipateur d'exergie. La ressource est la diminution d'exergie Δxh^+ du fluide qui le traverse (représentée par un lien entrant dans le port vert), et le produit est égal à 0.