Modélisation d'une pile SOFC dans Thermoptim

Une pile à combustible présente la particularité d'être traversée par deux flux séparés, qui échangent des ions ou des protons par l'intermédiaire d'un milieu imperméable pour les autres espèces chimiques. Elle se comporte donc comme un quadripôle recevant deux fluides en entrée, et dont en sortent deux autres.

Ceci pose une petite difficulté pour en réaliser un modèle dans Thermoptim, étant donné que les seuls composants disponibles sont soit des transfos, soit des nœuds. La solution est de former le quadripôle en associant un mélangeur en entrée et un diviseur en sortie, les deux étant reliés par une transfo-point qui joue un rôle purement passif.

Pour que le modèle soit bien cohérent, on synchronise les calculs effectués par les deux nœuds, plus précisément le diviseur de sortie prend le contrôle du mélangeur, dont le rôle se limite à effectuer une mise à jour des variables de couplage associées aux flux d'entrée. La structure du modèle est donnée figure 1.

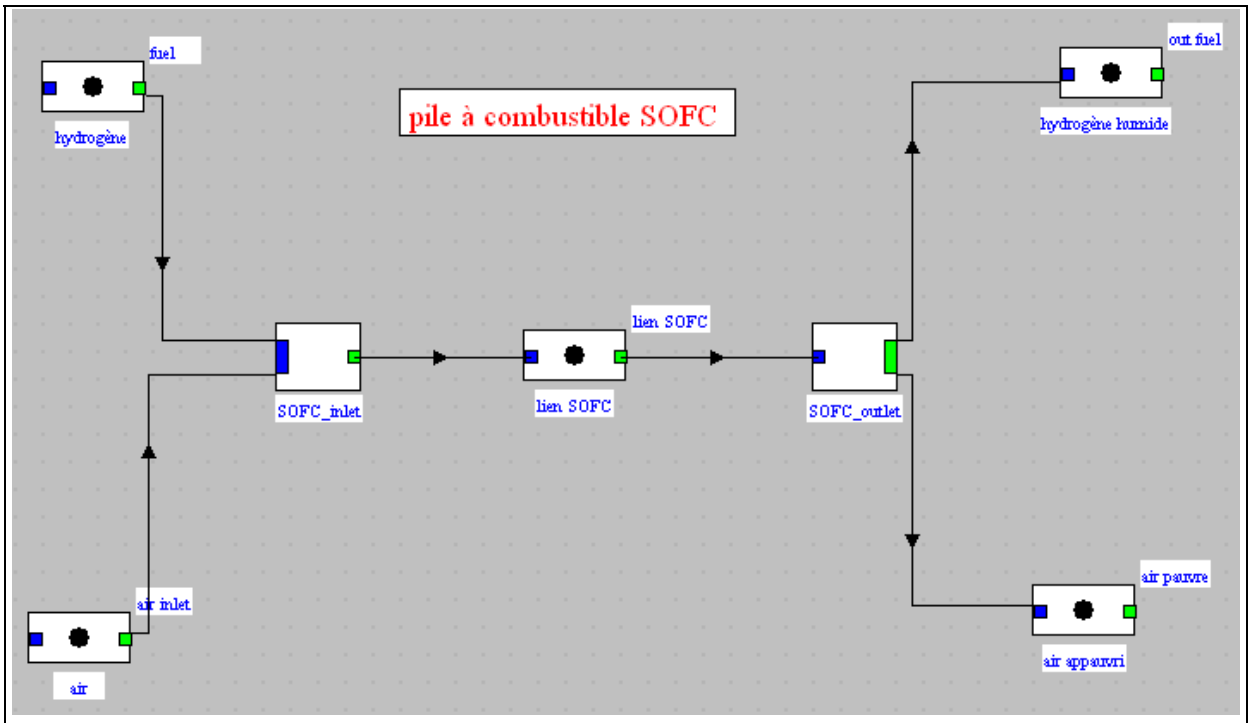


Figure 1 : Ecran du composant pile SOFC

Modèle de la pile

Le premier modèle que nous développerons est particulièrement simple : on suppose que le taux d'utilisation de l'hydrogène τ et le rendement énergétique ε sont connus.

Du côté de l'anode, seule la fraction τ est transformée dans la pile, le reste ressortant. Typiquement, $\tau = 0{,}5$. La réaction donnant les espèces en sortie s'écrit globalement :

Pour la partie utilisée : $\tau\ (H_2 + 1/2\ O_2) \rightarrow \tau\ H_2O$ ($\tau\ \Delta H0 = -\ 241\ 820\ \tau$ kJ/kmol)

Pour la partie non utilisée : $(1 - \tau)\ H_2$

Au total, on a donc en sortie d'anode : $(1 - \tau)\ H_2 + \tau\ H_2O$

$\tau\ \Delta H0$ représente l'énergie théorique mise en jeu par la conversion de la partie utilisée. Une fraction ε de cette énergie est directement convertie en électricité, et $(1 - \varepsilon)$ est transformée en chaleur.

Par ailleurs de l'oxygène est prélevé sur l'air comburant du côté de la cathode. λ représentant le flux d'air entrant, on a

air entrant : λ (O_2 + 3,76 N_2)

air appauvri sortant : (λ - τ/2) O_2 + 3,76 λ N_2

Le modèle que l'on peut retenir est alors le suivant :

1) la composition des espèces est donnée par la résolution des équations ci-dessus : on détermine les débits molaires d'hydrogène et d'air en entrée, ce qui fournit la valeur de λ, dont on déduit les débits molaires en sortie, les valeurs de τ et de ε étant lues à l'écran ;
2) la chaleur libérée par la fraction τ (1 - ε) du combustible sert à fournir l'énergie nécessaire à l'échauffement des gaz

L'enthalpie libérée est égale à $\dot{m}_{mol}\ \tau\ \Delta H0$. Elle se répartit entre de l'électricité $\dot{m}_{mol}\varepsilon\ \tau\ \Delta H0$, et de la chaleur pour l'échauffement des gaz $\dot{m}_{mol}\tau$ (1 - ε) $\Delta H0$, $\dot{m}_{mol}$ étant le débit molaire d'hydrogène.

Dans Thermoptim, la pile est comme on l'a dit représentée par un mélangeur externe connecté à un diviseur externe, les calculs étant effectués par ce dernier. Les classes s'appellent SOFCH2inlet et SOFCH2outlet.

L'écran du composant pile SOFC est donné figure 2. Les intitulés des données d'entrée du modèle sont affichés en bleu, et on a supposé que les gaz en entrée de pile étaient à la température de 500 °C.

Il s'agit d'une pile de technologie cylindrique du type de celle développée par Westinghouse, correspondant à un système de 100 kWe, pour laquelle τ = 0,48 et ε = 0,44.

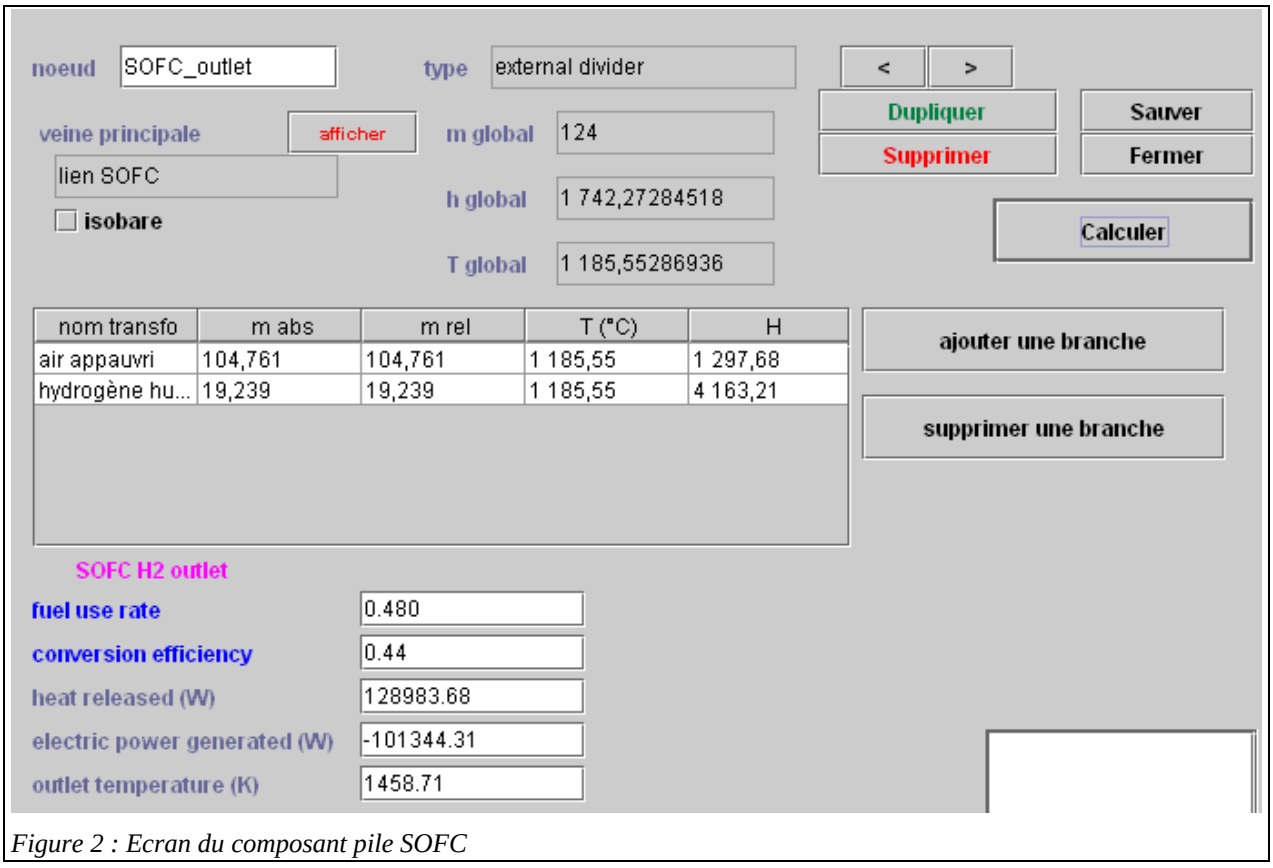


Figure 2 : Ecran du composant pile SOFC

Résultats du modèle simple

Avec les paramétrages ci-dessus (4 g/s d'hydrogène et 120 g/s d'air en entrée de la pile, à 1 bar et 500 °C), les compositions des gaz que l'on obtient sont les suivantes :

nom du composant	fraction molaire	fraction massique
N2	0,781	0,7555302
Ar	0,009	0,01241636
O2	0,21	0,2320534

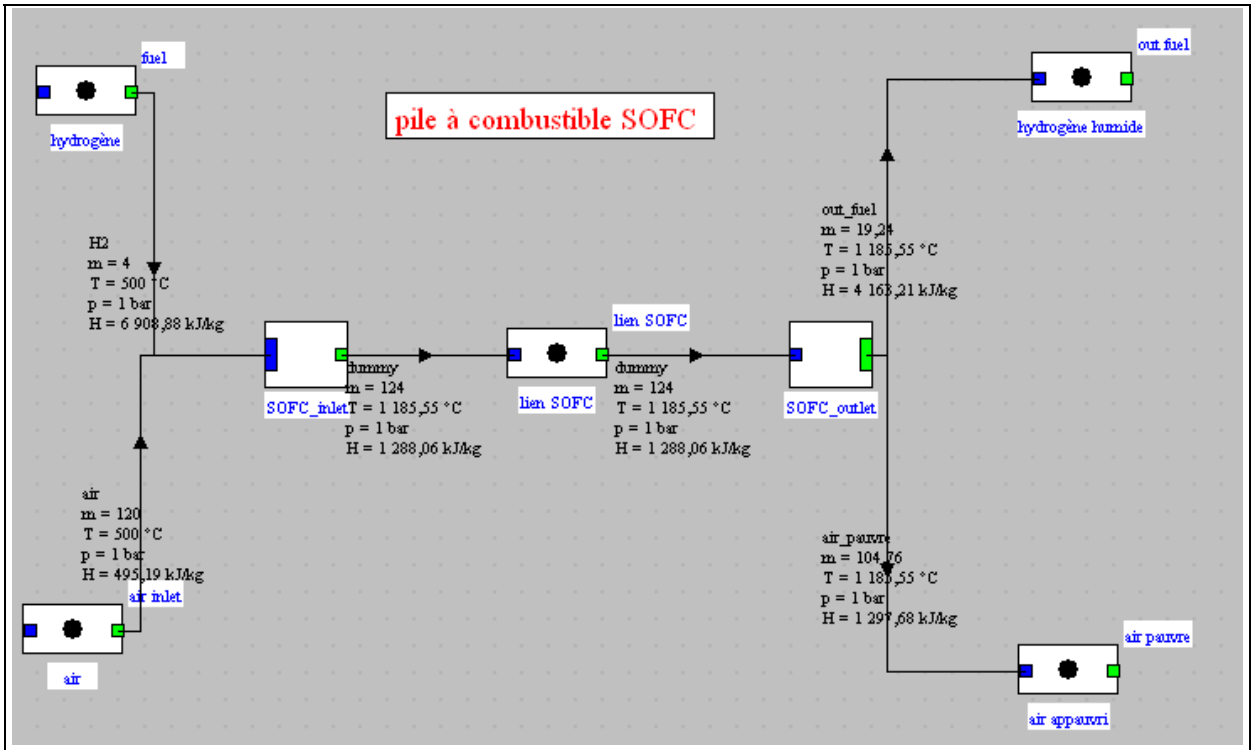
air en entrée de la pile, débit 120 g/s

nom du composant	fraction molaire	fraction massique
H2	0,48	0,09362063
H2O	0,52	0,9063794

hydrogène humidifié en sortie, débit 19,24 g/s, PCI : 11 230 kJ/kg

nom du composant	fraction molaire	fraction massique
N2	0,8824047	0,8654283
Ar	0,01016856	0,01422242
O2	0,1074267	0,1203493

air appauvri en O_2, débit 104,76 g/s



Ecran du composant pile SOFC

Avec ce modèle, diverses activités pédagogiques peuvent être proposées aux élèves :

- tout d'abord, modifier les valeurs du débit d'air entrant et de τ et ε, pour voir comment évoluent les performances de la pile
- insérer la pile dans un système plus complexe associant une turbine à gaz

- étudier comment le modèle est bâti, et éventuellement l'affiner en ajoutant par exemple un modèle électrique permettant de calculer τ et ε en fonction de l'intensité demandée
- modifier le modèle pour considérer le cas où la pile est refroidie par un fluide externe (ajout d'un thermocoupleur)
- modifier le modèle pour que la pile accepte un combustible autre que l'hydrogène pur (par exemple du méthane)

Etude de la classe externe SOFCH2outlet

Pour assurer la cohérence du modèle (éviter que l'on connecte le mélangeur d'entrée à un diviseur de sortie inadéquat), chacun des deux nœuds essaie d'instancier l'autre en recherchant sa classe parmi les composants externes du projet, et vérifie que tous deux sont bien connectés à la même transfo de liaison. Si l'opération échoue, un message avertit l'utilisateur que la construction est incorrecte. Cette vérification est effectuée par les méthodes setupOutletSOFC() et setupInletSOFC().

De surcroît, des tests de cohérence de chaque nœud sont effectués par la méthode checkConsistency() pour vérifier que les fluides connectés sont les bons : dans ce cas, de l'air et de l'hydrogène en entrée, et deux gaz en sortie, le repérage de l'air appauvri se faisant sur la présence d'azote. On se reportera au tome 3 du manuel de référence pour les explications sur ce point, valables pour tous les nœuds externes.

L'étude de la classe externe SOFCH2Outlet permet de voir comment le modèle a été implémenté. Comme on peut le voir, six étapes très simples suffisent pour effectuer les calculs :

1) les valeurs des paramètres τ et ε sont lues à l'écran :

```
epsi=getEpsilon(inletSOFC);//méthode de calcul de epsilon
tau=Util.lit_d(fuelUseRate_value.getText());
```

On fait ici appel à la méthode getEpsilon() pour prévoir le cas plus général où le rendement serait calculé et non lu à l'écran. Dans cet exemple, la méthode renvoie la valeur en double de ε.

2) les débits molaires en entrée sont obtenus du mélangeur externe d'entrée :

```
//débit molaire de H2
double molFlowH2=inletSOFC.fuelFlow/molMass[0];
//nombre de moles d'oxygène initial
double xO2=inletSOFC.fractO2;
double airMolFlow=inletSOFC.airFlow/inletSOFC.airM;

double O2MolFlow=airMolFlow*xO2;
double lambda=O2MolFlow/molFlowH2;
```

3) le débit molaire et la composition en sortie d'anode sont déterminés :

```
//nombre de moles du fuel en sortie
double xH2=1-tau;
double xH2O=tau;

Util.updateMolarComp(fuelComp,gasComp[0], xH2);
Util.updateMolarComp(fuelComp,gasComp[1], xH2O);

outletFuelSubstance.updateGasComposition(fuelComp);

double molFlow=molFlowH2*tau;
```

4) le débit molaire et la composition en sortie de cathode sont calculés

```
//débit molaire d'O2 consommé
double O2conso=molFlow/2.;

//débit restant d'O2
double O2restant=O2MolFlow-O2conso;
System.out.println("O2MolFlow : "+O2MolFlow+"  O2conso : "+O2conso+"  O2restant : "+O2restant);


//fraction du O2 initial utilisé
double epsiO2=(xO2-O2restant/airMolFlow)/xO2;
System.out.println("epsiO2 : "+epsiO2);
for(int i=0;i<inletSOFC.inletAirFractmol.length;i++){//mise à jour des fractions molaire de sortie
    double x_out=inletSOFC.inletAirFractmol[i]/(1-epsiO2*xO2);
    Util.updateMolarComp(airComp,inletSOFC.inletAirComp[i], x_out);
}
double xO2_out=(1-epsiO2)*xO2/(1-epsiO2*xO2);//fraction molaire de sortie en CO2
System.out.println("xO2_out : "+xO2_out);
Util.updateMolarComp(airComp,"O2", xO2_out);
outletAirSubstance.updateGasComposition(airComp);
```

5) les débits massiques sont calculés et mis à jour

```
//calcul des débits massiques en sortie
double volFlowAirPauvre=airMolFlow*inletSOFC.airM-O2conso*32;
System.out.println("volFlowAirPauvre : "+volFlowAirPauvre);
```

6) les calculs énergétiques sont enfin effectués

```
double DHO=-241830;//kJ/kmol H2

double Qlib=(tau*(-DHO)*(1-epsi))*molFlowH2;
double current=tau*DHO*epsi*molFlowH2;

System.out.println("Qlib : "+Qlib+"  current : "+current);
Q_value.setText(Util.aff_d(Qlib,2));
Current value.setText(Util.aff d(current,2));
```

7) la température de sortie déterminée, par dichotomie, en cherchant à annuler le résidu renvoyé par la fonction f_dicho (méthode Util.dicho_T). Le principe du calcul de la fonction f_dicho est le suivant : on cherche la température pour laquelle l'enthalpie totale des gaz en sortie de la pile (flowOutFuel*Hfuel+flowAirPauvre*Hsubst) est égale à la somme de l'enthalpie totale des gaz en entrée (inletSOFC.Htot) et de la puissance thermique dégagée par la pile Qlib.

```
double Htot=(inletSOFC.Htot+Qlib);///totalFlow;
System.out.println("Htot : "+Htot);
Taval=Util.dicho_T(this, 0, Htot, "outletTemp", inletSOFC.TamontAir, 2600, 0.01);
Tout_value.setText(Util.aff_d(Taval,2));
```

```
public double f_dicho(double T, double Htot,String fonc){
    if (fonc.equals("outletTemp")){
        double diff;
        Tpoint=T;
        outletFuelSubstance.CalcPropCorps(T,Ppoint, 1); //recalcul de l'état de sortie (fournit Hsubst)
        getSubstProperties(outletFuelSubstanceName);
        double Hfuel=Hsubst;
        outletAirSubstance.CalcPropCorps(T,Ppoint, 1); //recalcul de l'état de sortie (fournit Hsubst)
        getSubstProperties(outletAirSubstanceName);
        diff=Htot-flowOutFuel*Hfuel-flowAirPauvre*Hsubst;
        return diff;
    }
    return 0;
}
```

8) le nœud est mis à jour en utilisant les méthodes génériques décrites dans le manuel de référence