Electrothermal Simulation in GetDP with Newton's Method

Michael Asam - 21. February 2012

1. System of differential equations in strong form

Thermodynamic PDE:

$$\rho \cdot c(T) \cdot \frac{d}{dt} T + \nabla (-k(T) \cdot \nabla T) - q(\nabla V, T) = 0 \qquad \text{in } \Omega \tag{1.1}$$

with the heat source q

$$q(\nabla V, T) = \vec{J} \cdot \vec{E} \tag{1.2}$$

$$q(\nabla V, T) = \kappa(\nabla V, T) \cdot (\nabla V)^2 \tag{1.3}$$

Electric PDE:

$$\nabla (-\kappa(\nabla V, T) \cdot \nabla V) = 0 \qquad \text{in } \Omega \tag{1.4}$$

Electric natural boundary condition:

$$\kappa(\nabla V, T) \cdot \nabla V = J \qquad \text{at } \Gamma_{source} \tag{1.5}$$

2. System of differential equations in weak form

Thermodynamic PDE:

$$\int \rho \cdot c(T) \cdot \left(\frac{d}{dt} T\right) \cdot u \, d\Omega + \int k(T) \cdot \nabla T \cdot \nabla u \, d\Omega - \int \kappa(\nabla V, T) \cdot (\nabla V)^2 \cdot u \, d\Omega = 0 \tag{2.1}$$

Electric PDE:

$$\int \kappa(\nabla V, T) \cdot \nabla V \cdot \nabla w \, d\Omega - \int J \cdot w \, d\Gamma = 0 \tag{2.2}$$

u and w are so called test functions.

3. Newton-Raphson method

To solve the equations (2.1) and (2.2) the Newton-Raphson method is applied.
Therefore we define two functions:

$$Th(V,T) = \int \rho \cdot c(T) \cdot \left(\frac{d}{dt} T\right) \cdot u \, d\Omega + \int k(T) \cdot \nabla T \cdot \nabla u \, d\Omega - \int \kappa(\nabla V, T) \cdot (\nabla V)^2 \, d\Omega \qquad (3.1)$$

$$El(V,T) = \int \kappa(\nabla V, T) \cdot \nabla V \cdot \nabla w \, d\Omega - \int J \cdot w \, d\Gamma \qquad (3.2)$$

Target is to find V and T to set both functions equal zero.
The Newton-Raphson method acts according the following iteration scheme:

$$\begin{pmatrix} V_{i+1} \\ T_{i+1} \end{pmatrix} = \begin{pmatrix} V_i \\ T_i \end{pmatrix} - \begin{pmatrix} \frac{d}{dV} Th & \frac{d}{dT} Th \\ \frac{d}{dV} El & \frac{d}{dT} El \end{pmatrix}^{-1} \cdot \begin{pmatrix} Th(V_i, T_i) \\ El(V_i, T_i) \end{pmatrix} \qquad (3.3)$$

With

$$\begin{pmatrix} \delta V_{i+1} \\ \delta T_{i+1} \end{pmatrix} = \begin{pmatrix} V_{i+1} - V_i \\ T_{i+1} - T_i \end{pmatrix} \qquad (3.4)$$

equation (3.3) can be written in the following form:

$$Th(V_i, T_i) + \delta V_{i+1} \cdot \frac{d}{dV} Th(V_i, T_i) + \delta T_{i+1} \cdot \frac{d}{dT} Th(V_i, T_i) = 0 \qquad (3.5)$$

$$El(V_i, T_i) + \delta V_{i+1} \cdot \frac{d}{dV} El(V_i, T_i) + \delta T_{i+1} \cdot \frac{d}{dT} El(V_i, T_i) = 0 \qquad (3.6)$$

Those two equations have to be implemented in GetDP!

Note:
The unknowns are only δV_{i+1} and δT_{i+1}.
Everything else is known from the former iteration step.

For getting the derivatives of Th and El, equations (3.1) and (3.2) are splitted into five parts:

$$Th(V,T) = A(V,T) + B(V,T) + C(V,T) \tag{3.7}$$

$$A(V,T) = \int \rho \cdot c(T) \cdot \left(\frac{d}{dt}T\right) \cdot u \, d\Omega \tag{3.8}$$

$$B(V,T) = \int k(T) \cdot \nabla T \cdot \nabla u \, d\Omega \tag{3.9}$$

$$C(V,T) = -\int \kappa(\nabla V, T) \cdot (\nabla V)^2 \cdot u \, d\Omega \tag{3.10}$$

and

$$El(V,T) = F(V,T) + G(V,T) \tag{3.11}$$

$$F(V,T) = \int \kappa(\nabla V, T) \cdot \nabla V \cdot \nabla w \, d\Omega \tag{3.12}$$

$$G(V,T) = -\int J \cdot w \, d\Gamma \tag{3.13}$$

4. Newton-Raphson of thermal part

Inserting equations (3.7) .. (3.10) in (3.5) yields:

$$\begin{aligned} &A\left(V_i, T_i\right) + \delta V_{i+1} \cdot \frac{d}{dV} A\left(V_i, T_i\right) + \delta T_{i+1} \cdot \frac{d}{dT} A\left(V_i, T_i\right) \; ... \; = 0 \\ &+ B\left(V_i, T_i\right) + \delta V_{i+1} \cdot \frac{d}{dV} B\left(V_i, T_i\right) + \delta T_{i+1} \cdot \frac{d}{dT} B\left(V_i, T_i\right) \; ... \\ &+ C\left(V_i, T_i\right) + \delta V_{i+1} \cdot \frac{d}{dV} C\left(V_i, T_i\right) + \delta T_{i+1} \cdot \frac{d}{dT} C\left(V_i, T_i\right) \end{aligned} \tag{4.0.1}$$

4.1 Part A in GetDP-notation

Recall

$$A\left(V_i, T_i\right) = \int \rho \cdot c\left(T_i\right) \cdot \left(\frac{d}{dt} T_i\right) \cdot u \, d\Omega \qquad (4.1.1)$$

In GetDP-notation:

```
Galerkin{ [ rho[] * c[{T}] * Dt[{T}], {T} ]; … }
```
(4.1.2)

Where {T} referes to the temperature of the last iteration T_i.

The testfunction u is choosen to be equal T.

4.2 Linearisation of A with respect to V

Since A does not depend on V the derivative is zero:

$$\delta V_{i+1} \cdot \frac{d}{dV} A\left(V_i, T_i\right) = 0 \qquad (4.2.1)$$

Therefore there is no need to implement this therm in GetDP.

4.3 Linearisation of A with respect to T

For getting the derivative of A we need to take the time discrete equation for A:

$$A(V, T) = \int \rho \cdot c(T) \cdot \frac{T^n - T^{n-1}}{\Delta t} \cdot u \, d\Omega \qquad (4.3.1)$$

Here T^n is the temperature at the actual time point and T^{n-1} is the temperature one time step before. Δt is the time step.

The derivative of A is:

$$\frac{d}{dT}A(V,T) = \int \rho\cdot\left[\frac{c(T)}{\Delta t} + \left(\frac{d}{dT}c(T)\right)\cdot\frac{T^{n} - T^{n-1}}{\Delta t}\right]\cdot u\, d\Omega \qquad (4.3.2)$$

And therfore

$$\delta T_{i+1}\cdot\frac{d}{dT}A\left(V_i,T_i\right) = \int \rho\cdot\left(\frac{c\left(T_i\right)}{\Delta t} + \frac{d}{dT}c\left(T_i\right)\cdot\frac{d}{dt}T_i\right)\cdot\delta T_{i+1}\cdot u\, d\Omega \qquad (4.3.2)$$

In GetDP-notation:

```
dA_dT[ Region ] = rho[] * ( c[$1#1] / Delta_t + dc_dT[#1] * $2 );
```
(4.3.3)

```
Galerkin{ JacNL[ dA_dT[ {T}, Dt{T} ] * Dof{T}, {T} ]; … }
```
(4.3.4)

Here the new expression Dof{T} appears. In this case it corresponds to the unkown δT_{i+1}. The JacNL statement is to tell GetDP that this **D**egree **O**f **F**reedom is the delta per iteration and not the absolute value of the unknown temperature itself.

4.4 Part B in GetDP-notation

Recall

$$B\left(V_i,T_i\right) = \int k\left(T_i\right)\cdot\nabla T_i\cdot\nabla u\, d\Omega \qquad (4.4.1)$$

In GetDP-notation it would be written as

```
Galerkin{ [ k[{T}] * {d T}, {d T} ]; … }
```
(4.4.2)

But, as we will see in chapter 4.6, this part can be combined with an other in order to simplify the equations further and to save calculation time.

4.5 Linearisation of B with respect to V

Since B does not depend on V the derivative is zero:

$$\delta V_{i+1} \cdot \frac{d}{dV} B\left(V_i, T_i\right) = 0 \tag{4.5.1}$$

Therefore there is no need to implement this therm in GetDP.

4.6 Linearisation of B with respect to T

Recall:

$$B\left(V_i, T_i\right) = \int k\left(T_i\right) \cdot \nabla T_i \cdot \nabla u \, d\Omega \tag{4.6.1}$$

The linearisation is:

$$\delta T_{i+1} \cdot \frac{d}{dT} B\left(V_i, T_i\right) = \int k \cdot \left(\frac{d}{dT} \nabla T_i\right) \cdot \delta T_{i+1} \cdot \nabla u \, d\Omega + \int \left(\frac{d}{dT} k\right) \cdot \nabla T_i \cdot \delta T_{i+1} \cdot \nabla u \, d\Omega \tag{4.6.2}$$

To derive $\left(\frac{d}{dT} \nabla T_i\right) \cdot \delta T_{i+1}$ it is necessary to have a deeper look at :

$$T_i = \sum_{j=1}^{m} \left(T_{i,j} \cdot \alpha_j\right) \qquad \delta T_{i+1} = \sum_{j=1}^{m} \left(\delta T_{i+1,j} \cdot \alpha_j\right) \qquad (4.6.3) \quad (4.6.4)$$

Where j is the node number (there are m nodes) and α_j are the finite element basis functions for the thermal calculations. The gradient of this expression is

$$\nabla T_i = \sum_{j=1}^{m} \left(T_{i,j} \cdot \nabla \alpha_j\right) \tag{4.6.5}$$

$$\frac{d}{dT} \nabla T_i = \sum_{j=1}^{m} \nabla \alpha_j \tag{4.6.6}$$

$$\left(\frac{d}{dT}\nabla T_i\right)\cdot\delta T_{i+1} = \sum_{j=1}^{m}\left(\nabla\alpha_l\cdot\delta T_{i+1,j}\cdot\alpha_j\right) \tag{4.6.7}$$

The basis function α_j is equal 1 at each node and can be omitted at the end of the equation above.

$$\left(\frac{d}{dT}\nabla T_i\right)\cdot\delta T_{i+1} = \nabla\,\delta T_{i+1} \tag{4.6.8}$$

So finally we get:

$$\delta T_{i+1}\cdot\frac{d}{dT}B\left(V_i, T_i\right) = \int k\left(T_i\right)\cdot\nabla\,\delta T_{i+1}\cdot\nabla u\, d\Omega + \int \left(\frac{d}{dT}k\left(T_i\right)\right)\cdot\nabla\, T_i\cdot\delta T_{i+1}\cdot\nabla u\, d\Omega \tag{4.6.9}$$

As can be seen in equation (3.13) the tem (4.6.9) has to be added to several other terms. One of them is the term (4.4.1) which can be combined with with (4.6.8) in order to save computation effort.

$$\begin{aligned} B\left(V_i, T_i\right) + \delta T_{i+1}\cdot\frac{d}{dT}B\left(V_i, T_i\right) = & \int k\left(T_i\right)\cdot\nabla\, T_i\cdot\nabla u\, d\Omega\ ... \\ & + \int k\left(T_i\right)\cdot\nabla\,\delta T_{i+1}\cdot\nabla u\, d\Omega\ ... \\ & + \int \left(\frac{d}{dT}k\left(T_i\right)\right)\cdot\nabla\, T_i\cdot\delta T_{i+1}\cdot\nabla u\, d\Omega \end{aligned} \tag{4.6.10}$$

As defined above:

$$T_{i+1} = T_i + \delta T_{i+1} \tag{4.6.11}$$

Therefore (4.6.10) simplifies to

$$\begin{aligned} B\left(V_i, T_i\right) + \delta T_{i+1}\cdot\frac{d}{dT}B\left(V_i, T_i\right) = & \int k\left(T_i\right)\cdot\nabla\, T_{i+1}\cdot\nabla u\, d\Omega\ ... \\ & + \int \left(\frac{d}{dT}k\left(T_i\right)\right)\cdot\nabla\, T_i\cdot\delta T_{i+1}\cdot\nabla u\, d\Omega \end{aligned} \tag{4.6.12}$$

and in GetDP-notation:

```
Galerkin{          [ k[{T}] * Dof{d T}, {d T} ]; … }
```
(4.6.12)

```
Galerkin{ JacNL[ dk_dT[ {T} ] * {d T} * Dof{T}, {d T} ]; … }
```
(4.6.13)

4.7 Part C in GetDP-notation

Recall

$$C(V_i, T_i) = -\int \kappa(\nabla V_i, T_i)\cdot(\nabla V_i)^2 \cdot u \, d\Omega \tag{4.7.1}$$

In GetDP-notation it would be written as

```
q[ Region ] = ($1#11)^2 * kappa[ #1, $2 ];
```
(4.7.2)

```
Galerkin{ [ -q[{d V}, {T}], {T} ]; … }
```
(4.7.3)

4.8 Linearisation of C with respect to V

We have to find the derivative of C with respect to V. But unfortunately the terms in C depend only on ∇V. So we have to substitute the d/dV operator by d/d ∇V.

$$\delta V_{i+1}\cdot\frac{d}{dV}C(V_i, T_i) = \left(\frac{d}{d\nabla V}C(V_i, T_i)\right)\cdot\left(\frac{d}{dV}\nabla V_i\right)\cdot\delta V_{i+1} \tag{4.8.1}$$

The middle term of this equation can be derived in the following way:

$$\nabla V_i = \sum_{j=1}^{m}\left(V_{i,j}\cdot\nabla\beta_j\right) \qquad \delta V_{i+1} = \sum_{j=1}^{m}\left(\delta V_{i+1,j}\cdot\beta_j\right) \tag{4.8.2}$$

Where j is the node number (there are m nodes) and β_j are the finite element basis functions for the thermal calculations.

$$\frac{d}{dV}\nabla V_i = \sum_{j=1}^{m}\nabla\beta_j \tag{4.8.3}$$

$$\left(\frac{d}{dV}\nabla V_i\right)\cdot\delta V_{i+1} = \sum_{j=1}^{m}\left(\nabla\beta_j\cdot\delta V_{i+1}\cdot\beta_j\right) \tag{4.8.4}$$

The basis function β_j is equal 1 at each node and can be omitted at the end of the equation above.

$$\left(\frac{d}{dV}\nabla V_i\right)\cdot\delta V_{i+1} = \nabla\delta V_{i+1} \tag{4.8.5}$$

So finally we get:

$$\delta V_{i+1}\cdot\frac{d}{dV}C\left(V_i, T_i\right) = \int -\left[2\cdot\kappa\left(\nabla V_i, T_i\right)\cdot\nabla V_i + \left(\frac{d}{d\nabla V}\kappa\left(\nabla V_i, T_i\right)\right)\cdot\left(\nabla V_i\right)^2\right]\cdot\nabla\delta V_{i+1}\cdot u\, d\Omega \tag{4.8.6}$$

In GetDP-notation:

```
dq_dgradV[ Region ] = 2 * kappa[ $1#21, $2#22 ] * #21 +
                        dkappa_dgradV[ #21, #22 ] * #21^2;      (4.8.7)

Galerkin{ JacNL[ -dq_dgradV[{d V}, {T}] * Dof{d V}], {T} ]; ... }      (4.8.8)
```

Note:

The derivative $\frac{d}{d\nabla V}\kappa\left(\nabla V_i, T_i\right)$ gives in general a tensor because ∇V is a vector of the electric field.

$$\nabla V = -\vec{E} \tag{4.8.9}$$

$$\frac{d}{d\nabla V}\kappa = -\frac{d}{d\vec{E}}\kappa \tag{4.8.10}$$

$$\frac{d}{d\nabla V}\kappa\left(\nabla V_i, T_i\right) = -\begin{pmatrix} \frac{d}{dE_x}\kappa_x & \frac{d}{dE_y}\kappa_x & \frac{d}{dE_z}\kappa_x \\ \frac{d}{dE_x}\kappa_y & \frac{d}{dE_y}\kappa_y & \frac{d}{dE_z}\kappa_y \\ \frac{d}{dE_x}\kappa_z & \frac{d}{dE_y}\kappa_z & \frac{d}{dE_z}\kappa_z \end{pmatrix} \tag{4.8.11}$$

Where κ_x, κ_y and κ_z are the anisotrop electrical conductivities.

4.9 Linearisation of C with respect to T

Here we only need the derivative of κ:

$$\frac{d}{dT}C\left(V_i,T_i\right) = -\int \left(\nabla V_i\right)^2 \cdot \left(\frac{d}{dT}\kappa\left(\nabla V_i,T_i\right)\right)\cdot u\, d\Omega \tag{4.9.1}$$

So the linearisation is

$$\delta T_{i+1}\cdot\frac{d}{dT}C\left(V_i,T_i\right) = -\int \left(\nabla V_i\right)^2 \cdot \left(\frac{d}{dT}\kappa\left(\nabla V_i,T_i\right)\right)\cdot \delta T_{i+1}\cdot u\, d\Omega \tag{4.9.2}$$

In GetDP-notation:

```
dq_dT[ Region ] = ($1#31)^2 *dkappa_dT[ #31, $2#32 ];          (4.9.3)

Galerkin{ JacNL[ -dq_dT[{d V}, {T}] * Dof{T}], {T} ]; … }      (4.9.4)
```

5. Newton-Raphson of electrical part

Inserting equations (3.11) .. (3.13) in (3.6) yields:

$$\begin{aligned} &F\left(V_i,T_i\right) + \delta V_{i+1}\cdot\frac{d}{dV}F\left(V_i,T_i\right) + \delta T_{i+1}\cdot\frac{d}{dT}F\left(V_i,T_i\right) \ldots \quad = 0 \\ &+G\left(V_i,T_i\right) + \delta V_{i+1}\cdot\frac{d}{dV}G\left(V_i,T_i\right) + \delta T_{i+1}\cdot\frac{d}{dT}G\left(V_i,T_i\right) \end{aligned} \tag{5.0.1}$$

5.1 Part F in GetDP-notation

Recall

$$F\left(V_i,T_i\right) = \int \kappa\left(\nabla V_i,T_i\right)\cdot\nabla V_i\cdot\nabla w\, d\Omega \tag{5.1.1}$$

In GetDP-notation:

```
Galerkin{ [ kappa[ {d V}, {T} ] * {d V}, {d V} ]; … }      (5.1.2)
```

5.2 Linearisation of F with respect to V

The terms in C depend only on ∇V. Therefore we proceed in the same way as in section 4.8 and substitute the d/dV operator by d/d ∇V.

$$\delta V_{i+1} \cdot \frac{d}{dV} F\left(V_i, T_i\right) = \left(\frac{d}{d\nabla V} F\left(V_i, T_i\right)\right) \cdot \left(\frac{d}{dV} \nabla V_i\right) \cdot \delta V_{i+1} \tag{5.2.1}$$

Using equatin 4.8.5 we get

$$\delta V_{i+1} \cdot \frac{d}{dV} F\left(V_i, T_i\right) = \int \left(\kappa\left(\nabla V_i, T_i\right) + \nabla V_i \cdot \frac{d}{d\nabla V} \kappa\left(\nabla V_i, T_i\right)\right) \cdot \nabla \delta V_{i+1} \cdot \nabla w \, d\Omega \tag{5.2.2}$$

The electrical current density and the electrical field are given by

$$\vec{J}_i = -\kappa\left(\nabla V_i, T_i\right) \cdot \nabla V_i \tag{5.2.3}$$

$$\vec{E} = -\nabla V \tag{5.2.4}$$

The term in brackets in (5.2.2) is derivative

$$\frac{d}{d\vec{E}} \vec{J}_i = \kappa\left(\nabla V_i, T_i\right) + \nabla V_i \cdot \frac{d}{d\nabla V} \kappa\left(\nabla V_i, T_i\right) \tag{5.2.5}$$

Inserting (5.2.5) in (5.2.2) yields

$$\delta V_{i+1} \cdot \frac{d}{dV} F\left(V_i, T_i\right) = \int \left(\frac{d}{d\vec{E}} \vec{J}_i\right) \cdot \nabla \delta V_{i+1} \cdot \nabla w \, d\Omega \tag{5.2.6}$$

In GetDP-notation:

```
Galerkin{ JacNL[ dJ_dE[ {d V}, {T} ] * Dof{d V}, {d V} ]; … }
```
(5.2.7)

Note:

The derivative $\frac{d}{d\vec{E}}\vec{J}_i$ is a tensor.

$$\frac{d}{d\vec{E}}\vec{J}_i = \begin{pmatrix} \frac{d}{dE_x}J_x & \frac{d}{dE_y}J_x & \frac{d}{dE_z}J_x \\ \frac{d}{dE_x}J_y & \frac{d}{dE_y}J_y & \frac{d}{dE_z}J_y \\ \frac{d}{dE_x}J_z & \frac{d}{dE_y}J_z & \frac{d}{dE_z}J_z \end{pmatrix} \tag{5.2.8}$$

In the 3D DMOS area the current flows only in z-direction. Therefore all elements of the tensor but $\frac{d}{dE_z}J_z$ are equal 0.

$$\frac{d}{d\vec{E}}\vec{J}_i = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{d}{dE_z}J_z \end{pmatrix} \tag{5.2.9}$$

$$\frac{d}{dE_z}J_z = \kappa\left(\nabla_z V_i, T_i\right) + E_z \cdot \frac{d}{dE_z}\kappa\left(\nabla_z V_i, T_i\right) \tag{5.2.10}$$

In GetDP-notation:

```
dJ_dE[ Region ] = kappa[ CompZ[$1]#41, $2#42 ] + #41 * dkappa_dE[ #41, #42]
```
(5.2.11)

5.3 Linearisation of F with respect to T

Here we only need the derivative of κ:

$$\frac{d}{dT}F(V_i, T_i) = \int \left(\frac{d}{dT}\kappa(\nabla V_i, T_i)\right) \cdot \nabla V_i \cdot \nabla w \, d\Omega \tag{5.3.1}$$

So the linearisation is

$$\delta T_{i+1} \cdot \frac{d}{dT}F(V_i, T_i) = \int \left(\frac{d}{dT}\kappa(\nabla V_i, T_i)\right) \cdot \nabla V_i \cdot \delta T_{i+1} \cdot \nabla w \, d\Omega \tag{5.3.2}$$

In GetDP-notation:

```
dq_dT[ Region ] = ($1#31)^2 *dkappa_dT[ #31, $2#32 ];                       (5.3.3)

Galerkin{ JacNL[ dkappa_dT[{d V}, {T}] * {d V} * Dof{T}], {d V} ]; … }     (5.3.4)
```

5.4 Part G in GetDP-notation

Recall

$$G(V_i, T_i) = -\int J \cdot w \, d\Gamma \tag{5.4.1}$$

In GetDP-notation:

```
Galerkin{ [ -J[], {V} ]; … }     (5.4.2)
```

5.5 Linearisation of G

The term G does not depend on V or T. Therefore all derivatives of G are zero.