Further Applications of the PDE Solver FDEM with Error Estimate to Different PDE Systems

Torsten Adolph and Willi Schönauer

Forschungszentrum Karlsruhe, Institut für Wissenschaftliches Rechnen,
Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany
{torsten.adolph,willi.schoenauer}@iwr.fzk.de

Summary. We give a brief overview of the Finite Difference Element Method (FDEM), above all how an explicit estimate of the error is obtained. Then for some academic examples, the estimated and exact error are compared showing the quality of the estimate. The nonlinear PDE for the numerical simulation of the temperature in a high pressure Diesel injection pump, where the domain consists of three subdomains, is solved. Then the temperature in a power semiconductor module with six power chips is numerically simulated. This is a time-dependent problem in 3-D, where the solution domain consists of two subdomains. For both problems, the global error estimate shows the quality of the solution, and it would be very difficult to obtain a quality control of the solution by conventional grid refinement tests.

1 Introduction

In this paper, we present the results of two problems that we solved in cooperation with industrial and academic partners: the numerical simulation of the temperature in a high pressure Diesel injection pump and of a power semiconductor module with six power chips.

Never before such problems have been solved with error estimates. So the emphasis of this paper will be on the error estimate: together with the solution we present values or plots for the error estimates. Because of the limited accorded space of the paper, we cannot present all the details of FDEM. However, we will give the precise information where these details are in the corresponding report [1] which can be accessed in the Internet.

2 The Finite Difference Element Method (FDEM)

FDEM is an unprecedented generalization of the FDM on an unstructured FEM mesh. It is a black-box solver for arbitrary nonlinear systems of 2-D and 3-D elliptic or parabolic PDEs. If the unknown solution is $u(t, x, y, z)$ the operator for PDEs and BCs (boundary conditions) is (2.4.1) and (2.4.2) in [1]:

$$Pu \equiv P(t,x,y,z,u,u_t,u_x,u_y,u_z,u_{xx},u_{yy},u_{zz},u_{xy},u_{xz},u_{yz}) = 0 \ . \tag{1}$$

For a system of m PDEs, u and Pu have m components:

$$u = \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix} \ , \quad Pu = \begin{pmatrix} P_1 u \\ \vdots \\ P_m u \end{pmatrix} \ . \tag{2}$$

As we have a black-box solver, the PDEs and BCs and their Jacobian matrices of type (2.4.6) in [1] must be entered as Fortran code in prescribed frames.

The geometry of the domain of solution is entered as a FEM mesh with triangles in 2-D and tetrahedra in 3-D. The domain may be composed of subdomains with different PDEs and non-matching grid. From the element list and its inverted list, we determine for each node more than the necessary number of nodes for difference formulas of a given consistency order q. By a sophisticated algorithm, from this set the necessary number of nodes is selected, see Sect. 2.2 in [1]. From the difference of formulas of different consistency order, we get an estimate of the discretization error. If we want e.g. the discretization error for u_x, and $u_{x,d,q}$ denotes the difference formula of consistency order q, the error estimate d_x is defined by

$$d_x := u_{x,d,q+2} - u_{x,d,q} \ , \tag{3}$$

i.e. by the difference to the order $q+2$. This has a built-in self-control: if this is not a "better" formula the error estimate shows large error.

With such an error estimate, we can explicitly compute the error of the solution by the error equation (2.4.8) in [1]. The knowledge of the error estimate allows a mesh refinement and order control in space and time (for parabolic PDEs), see Sect. 2.5 in [1].

A special problem for a black-box solver is the efficient parallelization because the user enters his domain by the FEM mesh. We use a 1-D domain decomposition with overlap to distribute the data to the processors, see Sect. 2.8 in [1]. We use MPI. A detailed report on the parallelization is [2]. The resulting large and sparse linear system is solved by the LINSOL program package [3] that is also efficiently parallelized for iterative methods of CG type and (I)LU preconditioning.

3 Academic Examples

The purpose of these academic examples is to check the quality of the error estimate (which is one level higher than the usual check for the quality of the solution), see Sect. 2.10 in [1]. We define the global relative error for a component l of the solution and the global relative error by

$$\frac{\|\Delta u_{d,l}\|}{\|u_{d,l}\|} \ , \quad \frac{\|\Delta u_d\|}{\|u_d\|} = \max_l \frac{\|\Delta u_{d,l}\|}{\|u_{d,l}\|} \ , \tag{4}$$

where $\Delta u_{d,l}$ is computed from the error equation (component l of (2.4.8) in [1]). The norm $\|\cdot\|$ is the maximum norm. We generate from the original PDE $Pu = 0$ a "test PDE" for a given solution $\bar{u}$ by $Pu - P\bar{u} = 0$ that has $\bar{u}$ as solution. This prescription holds also for the BCs. The exact global relative error then is

$$\frac{\|\bar{u} - u_d\|}{\|u_d\|} \,. \tag{5}$$

We compute on the HP XC6000 with Itanium2 processors of 1.5 GHz (University of Karlsruhe). As exact solution $\bar{u}$ we select either a polynomial of a given order or a sugar loaf type function (2.10.16) in [1]. We solve the Navier-Stokes equations in velocity/vorticity form (2.10.13) in [1] with the unknown functions velocity components u, v and vorticity ω, and Reynolds number $Re = 1$. We solve on a circle with radius $= 1$ on a grid with 751 nodes, 1410 elements that has been generated by the commercial mesh generator I-DEAS. We compute with 8 processors. The given CPU time is that of the master processor 1.

Table 1. Results for the solution of the Navier-Stokes type equations on a circle with 751 nodes for different consistency orders q and test function $\bar{u}$

	order $q = 2$		order $q = 4$		order $q = 6$	
type $\bar{u}$	error exact error estim.	CPU sec.	error exact error estim.	CPU sec.	error exact error estim.	CPU sec.
pol. order 6	0.154 0.155	0.158	$0.914 \cdot 10^{-2}$ $0.367 \cdot 10^{-1}$	0.175	$0.108 \cdot 10^{-10}$ $0.109 \cdot 10^{-8}$	2.131
sugar loaf	$0.694 \cdot 10^{-1}$ $0.642 \cdot 10^{-1}$	0.168	$0.238 \cdot 10^{-1}$ $0.220 \cdot 10^{-1}$	0.184	$0.457 \cdot 10^{-2}$ 0.736^{*}	1.853

* here the order 8 for the error estimate is overdrawn (too coarse grid)

Table 1 shows the results. Here are two remarks: for $\bar{u}$ polynomial of order 6 and consistency order $q = 6$ we should reproduce $\bar{u}$ exactly which is expressed by the small errors. For the sugar loaf function and consistency order $q = 6$, we get a large error estimate. This shows the built-in self-control: near the top of the sugar loaf the grid is too coarse for the consistency order $q + 2 = 8$ that is used for the error estimate, the order 8 is "overdrawn" (higher order may not be better).

For the demonstration of the self-adaptation, we solve the same problem with the sugar loaf function again with 8 processors, but now we switch on the mesh refinement and order control for a global relative error of 0.25%. The results are shown in Table 2. The requested accuracy needs three refinement cycles. "no. of nodes ref." is the number of refinement nodes that determine the refinement elements from which then follows the new number of nodes. Observe the excellent error estimate that results from the optimal local order. Figure 1 shows the grid after the 3^{rd} cycle, the refinement is clearly visible.

Table 2. Results for the self-adaptation of mesh and order for sugar loaf test function for prescribed global relative error $0.25 \cdot 10^{-2}$ (0.25%)

cycle	no. of nodes	no. of elem.	no. of nodes ref.	no. of nodes with order 2	4	6	global relat. error exact	estimated	sec. for cycle
1	751	1410	132	427	320	4	$0.305 \cdot 10^{-1}$	$0.280 \cdot 10^{-1}$	1.021
2	1332	2493	345	180	1144	8	$0.109 \cdot 10^{-1}$	$0.950 \cdot 10^{-2}$	3.604
3	2941	5469	–	360	2556	25	$0.179 \cdot 10^{-2}$	$0.174 \cdot 10^{-2}$	10.086

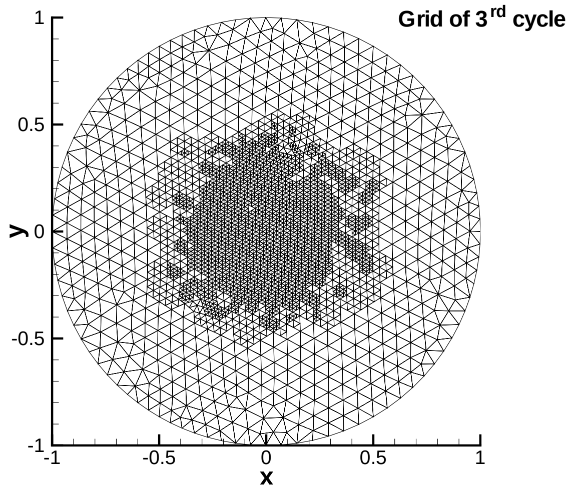

Fig. 1. Refined grid after 3rd cycle of Table 2

As mentioned above we wanted to demonstrate the quality of the error estimate. The estimate is the better the smaller the error is, this is a natural consequence of (3). What we have seen in Table 1 is also part of our test technique for each new problem: at first we create from the new problem a test PDE $Pu - P\bar{u} = 0$ and check with polynomial test solutions $\bar{u}$ the error estimate like in Table 1.

4 Heat Conduction in a Thin Annulus

Based on a former research project (see [1, Sect. 3.3, p. 110], [4]), we are interested in the additional calculation of the stationary temperature field of a model fluid in a very thin annulus between a housing and a piston.

The domain consists of three subdomains with different systems of PDEs: piston, housing and fluid. In the former research project, we computed the stresses and the displacements w and u in z- and r-direction in the piston and the housing, and the pressure p and the velocities w and u in z- and r-direction of the fluid in the lubrication gap.

By the pressure of 2000 bar, the lubrication gap widens and changes its form. For the actual problem, this changed form and the velocities in the lubrication gap are given, and by the solution of the PDE systems we obtain the temperature sequence in piston, housing and, as a matter of particular interest, in the gap.

The heat equation for an incompressible Newtonian fluid using axisymmetric cylindrical coordinates is:

$$u\,\frac{\partial T}{\partial r} + w\,\frac{\partial T}{\partial z} = \kappa\left(\frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} + \frac{\partial^2 T}{\partial z^2}\right) + 2\,\frac{\nu}{c_p}\hat{\varepsilon}^2 \tag{6}$$

with

$$\hat{\varepsilon}^2 = \left(\frac{\partial u}{\partial r}\right)^2 + \left(\frac{\partial w}{\partial z}\right)^2 + \frac{u^2}{r^2} + \frac{1}{2}\left(\frac{\partial w}{\partial r} + \frac{\partial u}{\partial z}\right)^2 . \tag{7}$$

Therein, $T = T(r,z)$ is the unknown temperature, u and w are the velocities known from the solution of the fluid-structure interaction problem, $\hat{\varepsilon}$ is the strain tensor, κ is the thermal diffusivity, ν is the kinematic viscosity and c_p is the specific heat capacity.

In order to distinguish between quantities in the fluid and quantities in the piston and the housing respectively, we denote quantities in the piston and in the housing with an asterisk. Due to $u^* = w^* = 0$, the heat equation is reduced to:

$$\frac{\partial^2 T^*}{\partial r^2} + \frac{1}{r}\frac{\partial T^*}{\partial r} + \frac{\partial^2 T^*}{\partial z^2} = 0 . \tag{8}$$

Both piston and housing consist of steel, while the lubricant in the gap is a model fluid. The material properties for steel and fluid are given in Table 3.

Table 3. Material parameters for the model fluid and steel

κ	$=$	$91.5 \cdot 10^{-9}\,\mathrm{m^2/s}$,	κ^*	$=$	$10.7 \cdot 10^{-6}\,\mathrm{m^2/s}$
λ	$=$	$0.15\,\mathrm{W/(m\,K)}$,	λ^*	$=$	$42\,\mathrm{W/(m\,K)}$
ϱ	$=$	$800\,\mathrm{kg/m^3}$,	ϱ^*	$=$	$7850\,\mathrm{kg/m^3}$
c_p	$=$	$2050\,\mathrm{J/(kg\,K)}$,	c_p^*	$=$	$502\,\mathrm{J/(kg\,K)}$
η	$=$	$2 \cdot 10^{-3}\,\mathrm{Pa\,s}$	ν	$=$	$2.5 \cdot 10^{-6}\,\mathrm{m^2/s}$

Therein, ϱ and ϱ^* are the densities, η is the dynamic viscosity and λ, λ^* are the heat conductivities. Furthermore, it holds

$$\nu = \frac{\eta}{\varrho}\,, \qquad \kappa = \frac{\lambda}{\varrho\, c_p}\,. \tag{9}$$

On the outer boundaries at $z = 0$ and $r = r_a$, respectively, we assume

$$T = T^* = T_0 := 20℃ . \tag{10}$$

On the symmetry axis at $r = 0$, it holds

$$\frac{\partial T^*}{\partial r} = 0 . \tag{11}$$

On the outer boundary at $z = z_e$, we assume

$$\frac{\partial T}{\partial z} = \frac{\partial T^*}{\partial z} = 0 . \tag{12}$$

At the fluid-steel interfaces we postulate the following two conditions:

$$T^* = T , \qquad \lambda^* \frac{\partial T^*}{\partial r} = \lambda \frac{\partial T}{\partial r} . \tag{13}$$

Figure 2 shows the 8 external boundaries and the 2 dividing lines (DL) of the domain. The boundary conditions at the 12 corners of the three subdomains are shown in Table 4.

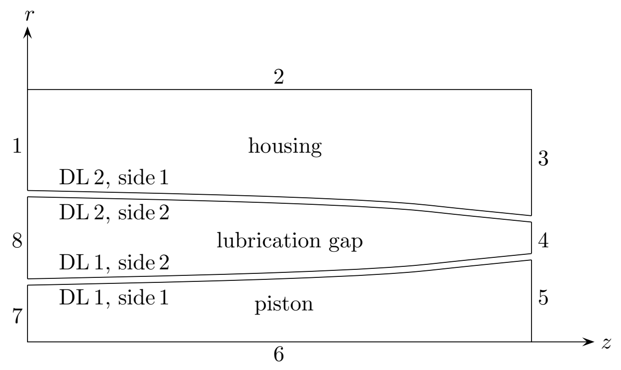

Fig. 2. Illustration of external boundaries and dividing lines (DL) of piston, lubrication gap and housing. The lubrication gap that has a width of a few micrometers is largely blown up

For the computations, we have used the distributed memory supercomputer HP XC6000 of the University of Karlsruhe with Itanium2 processors, 1.5 GHz, 2-processor nodes with Quadrics interconnect. We computed in parallel on 16 processors. The grids for the three subdomains are:

piston:	401 (z-direction)$\times$40 (r-direction),
lubrication gap:	401×641,
housing:	401×81.

Table 4. Boundary conditions for the corners of the 3 subdomains

subdomain	upper left	upper right	lower left	lower right
piston	$T^* = T_0 = 20$	$T^* = T$	$T^* = T_0 = 20$	$\frac{\partial T^*}{\partial z} = 0$
lubrication gap	$T = T_0 = 20$	$\frac{\partial T}{\partial z} = 0$	$T = T_0 = 20$	$\frac{\partial T}{\partial z} = 0$
housing	$T^* = T_0 = 20$	$T^* = T_0 = 20$	$T^* = T_0 = 20$	$T^* = T$

The computation time for the master processor 1 is 155 sec. The results of the computation are shown in Table 5 where we present the maximum temperature, the maximum relative estimated error and the mean relative estimated error for the three subdomains.

Table 5. Maximum temperature, maximum and mean relative estimated error for piston, lubrication gap and housing

		rel. estim. error	
subdomain	T_{max} (°C)	max.	mean
piston	98.2	$0.29 \cdot 10^{-1}$	$0.46 \cdot 10^{-4}$
lubrication gap	98.9	$0.29 \cdot 10^{-1}$	$0.84 \cdot 10^{-3}$
housing	87.8	$0.40 \cdot 10^{-2}$	$0.24 \cdot 10^{-4}$

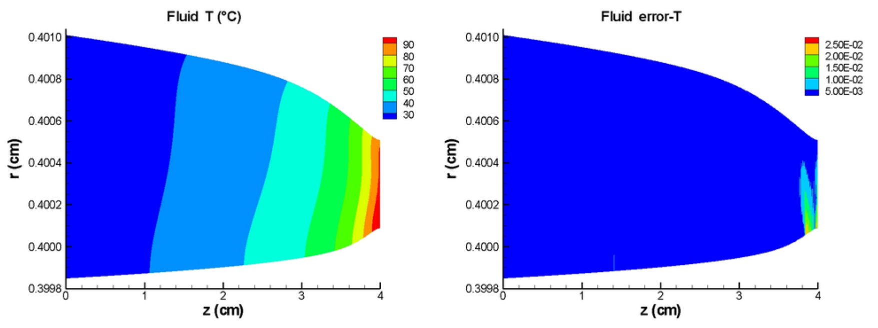

Fig. 3. Contour plot for the temperature T and its error in the lubrication gap

We see that the maximum relative errors are about 3% for the piston and the lubrication gap, but errors in the range of the maximum error appear only in a few nodes as the mean relative errors are $0.46 \cdot 10^{-4}$ in the piston and $0.84 \cdot 10^{-3}$ in the lubrication gap. In the housing the errors are even smaller.

Figure 3 shows the temperature T in the fluid and its error. The temperature increases from 20°C at $z = 0$ to 98.9°C at $z = 4$ cm. From the error picture at the right side of the figure we can also see that the maximum errors occur only in few nodes. For the contour plots of the temperature in the piston and the housing, we refer to [5].

5 Simulation of a Power Semiconductor Module

In the following a thermal problem will be presented which is encountered in the thermal predictive simulation of power semiconductor modules (e.g. dc/ac-converters). Heat sources are MOSFET-devices (or other semiconductor devices) on the top side of the module. The cooling is applied at the bottom side of the module, either by a convective liquid or gas (air) stream. The temperature evolution $T(x,t)$ in the module and in the power dissipating devices is described by the heat conduction equation:

$$\left(\varrho(x)c(x)\frac{\partial}{\partial t} - \nabla_x \cdot (\lambda(x)\nabla_x)\right) T(x,t) = H(x,t,T(x,t)) \ . \tag{14}$$

x denotes the 3-D position (x,y,z). $H(x,t,T(x,t))$ is the heat generation density (W/cm^3) in the system, which will generally depend on the temperature $T(x,t)$ of the heat source at position and time (x,t), because e.g. on-state and switching losses of the MOSFETs are temperature dependent. ϱ, c and λ denote the local mass density, the specific heat and the thermal conductivity, respectively. Strictly speaking, ϱ, c and λ also depend on the local temperature. However, it turns out that in power semiconductor modules material parameters for constant mean temperatures can be used in good approximation. So it holds

$$\varrho c \frac{\partial T}{\partial t} - \lambda\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right) - H = 0 \ . \tag{15}$$

An essential nonlinearity arises due to the convective cooling at the bottom side, which is included by a corresponding boundary condition:

$$\lambda(x)\frac{\partial T(x,t)}{\partial n} = \alpha(x,T,T_a)(T(x,t) - T_a(x,t)) + J(x,t) \ . \tag{16}$$

In (16), $\alpha(x,T,T_a)$ denotes a local heat-transfer coefficient at an edge point x of the structure with local temperature $T(x,t)$ and local ambient (fluid-)temperature $T_a(x,t)$. $\partial/\partial n$ denotes the normal derivative with outside direction. $J(x,t)$ is an optional surface heat source that is put to zero in our case.

For a simplified model, only cooling at the bottom side is assumed, while the other sides are thermally adiabatic. When the module is mounted on an air cooled radiator, the effective cooling area of the module is increased considerably, e.g. by a factor of 100. Simplified, (15) can be written as

$$-\lambda(x)\frac{\partial T}{\partial z} + \sigma \cdot \left(T^4 - T_a^4\right) + a \cdot (T - T_a)^{5/4} = 0 \tag{17}$$

with σ, a and T_a as given in Table 6. The sidewall and topside boundary conditions of the module are adiabatic ($\partial T/\partial n = 0$).
In order to have simple geometry and boundary conditions the whole module is assumed to be of rectangular structure with uniform material. The dimensions (side lengths) of the module are: 11.8 cm length; 5.8 cm width; 0.5 cm height.

There are six chip heat sources in the module which do not disturb the geometry of the rectangular parallelepiped and are assumed as embedded regions with heat generation density $H(x,t)$ different from zero. Those six "chips" are quadratic and of equal dimension ($0.9 \times 0.9\,\mathrm{cm}^2$) oriented in one row at the module topside. The chip thickness is 0.02 cm (200 µm). The distances of the chips left edge from the left side of the module are in (cm): $x_1 = 1.35$, $x_2 = 2.95$, $x_3 = 4.55$, $x_4 = 6.15$, $x_5 = 7.75$, $x_6 = 9.35$ (in the distance of 1.6 cm). The chip distance in y-direction from the module side wall is 1.9 cm. The chips topside is on the same plane as the rectangular module top-side.

The following calculations are performed:

1. At start time $t = 0$ the module has homogeneous temperature $T_a = 297\,\mathrm{K}$. The chips are turned on with power dissipation of 250 W/Chip. This means a heat generation density H of $250\,\mathrm{W/chipvolume} = 15432.1\,\mathrm{W/cm}^3$ for all six chips. What are the temperature contours on the module topside after 50 seconds?
2. Additionally to problem 1), there is a degraded array of MOSFET cells with lateral dimensions of $0.1 \times 0.1\,\mathrm{cm}^2$ with considerably increased power dissipation in the 6^{th} chip from the left. The chips 1–5 are heated again with constant 250 W (starting at $t = 0$), as is chip 6. However, now in chip 6 there is an additional 50 W of dissipated power in the degraded cell array, so that the total power of chip 6 is 300 W. The location of the quadratic degraded cell array is in the distance of 0.65 cm in x- and y-direction from the left lower corner of chip 6. The thickness of the extra heating region is as usual 0.02 cm. Temperature contours as for problem 1) would be of interest.

This is a 3-D problem, and as we want to compute the temperature distribution on the surface of the module at a given time, it is also time dependent. Furthermore, as the MOSFET-devices are very thin in comparison with the remainder of the module, we separate the module into two subdomains: the upper subdomain is as thick as the MOSFET-devices, the lower subdomain contains the remainder of the module. As we expect greater temperature gradients in the upper subdomain, and as it would be too costly to have the same fine grid in the lower subdomain, we choose different mesh sizes in x- and y-direction in the two subdomains. Thus, we introduce a sliding dividing line (SDL) between the two subdomains that allows for non-matching grid; they are coupled by coupling conditions, see [1, Sect. 2.6].

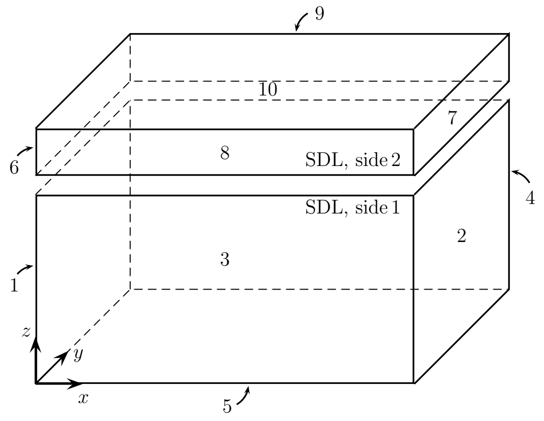

Fig. 4. Illustration of the subdomains with external boundaries and SDL

Figure 4 shows the 10 external boundaries and the SDL of the domain. The coupling conditions on the SDL between the two subdomains are:

$$T_{upper} = T_{lower} \ , \qquad \frac{\partial T_{upper}}{\partial z} = \frac{\partial T_{lower}}{\partial z} \ . \tag{18}$$

In Table 6 we give the material parameters for the solid materials that we use for the computation.

Table 6. Material parameters

λ	$=$	$1.51\,\mathrm{W/(K\,cm)}$,	ϱ	$=$	$2.32\,\mathrm{g/cm^3}$
c	$=$	$0.851\,\mathrm{Ws/(g\,K)}$,	T_a	$=$	$297\,\mathrm{K}$
a	$=$	$0.406 \cdot 10^{-1}\,\mathrm{W/(cm^2\,K^{5/4})}$,	σ	$=$	$0.567 \cdot 10^{-9}\,\mathrm{W/(cm^2\,K^4)}$
J	$=$	$0\,\mathrm{W/cm^2}$			

We carried out the computations on the distributed memory supercomputer HP XC4000 with AMD Opteron processors, 2.6 GHz, and InfiniBand interconnect that has been installed at the University of Karlsruhe. We computed in parallel on 32 processors, and we used the consistency orders $q = 2$ and $q = 4$ to see the influence of the order. We used $237 \times 117 \times 9 = 249{,}561$ nodes in the upper subomain, and $119 \times 59 \times 9 = 63{,}189$ nodes in the lower subdomain. Therefore, the total number of grid points is 312,750. For the mesh size h_x in x-direction and the mesh size h_y in y-direction, it holds $h_x = h_y$ in the lower and the upper subdomain, but we get for the ratio of the mesh sizes in the two subdomains $h_{x,u}/h_{x,l} = h_{y,u}/h_{y,l} = 1/2$. The ratio of the mesh sizes in the z-direction in the two subdomains is $h_{z,u}/h_{z,l} = 1/24$.

For the first problem, all six power chips have the same power dissipation of 250 W/Chip, so it holds for the heat generation density H

$$H = 250\,\mathrm{W/chipvol.} = 250\,\mathrm{W}/(0.9 \cdot 0.9 \cdot 0.02)\mathrm{cm}^3 = 15432.1\,\mathrm{W/cm}^3 \ . \quad (19)$$

In Table 7 we present the results of the first problem where all six chips have the same power dissipation. For the consistency orders $q = 2$ and $q = 4$, you can see the maximum temperature T_{max} for each of the two subdomains and the errors of the solution. The maximum error is the maximum of the global relative estimated error, i.e. it is the maximum absolute error in the subdomain divided by the maximum of the temperature. The mean error is the arithmetic mean of all relative errors in the subdomain. The given CPU time is that of the master processor 1.

Table 7. Results of the first calculation with $H = 250\,\mathrm{W/Chip}$

			rel. estim. error		CPU
Order	Subd.	T_{max} (K)	max.	mean	time (h)
$q = 2$	upper	526.4	$0.11 \cdot 10^{-1}$	$0.81 \cdot 10^{-3}$	7.0
	lower	524.2	$0.69 \cdot 10^{-2}$	$0.17 \cdot 10^{-3}$	
$q = 4$	upper	526.2	$0.78 \cdot 10^{-2}$	$0.17 \cdot 10^{-3}$	42.0
	lower	524.1	$0.22 \cdot 10^{-2}$	$0.35 \cdot 10^{-4}$	

For order $q = 2$, the maximum temperature, which is actually met in the centre of the 3^{rd} chip from the left, is 526.4 K after 50 sec. In the lower subdomain, the maximum temperature is only slightly smaller. The maximum error in the upper domain is about 1%, in the lower subdomain it is only 0.7%, which means that we have a solution that is accurate to 6 K. The mean errors are much smaller which means that the maximum errors occur only in few nodes. Considering the mean error, the solution is accurate to 0.5 K.

We see that the maximum temperatures for order $q = 2$ and $q = 4$ differ only slightly, but the maximum errors in the subdomains are reduced to 2/3 in the upper and to 1/3 in the lower subdomain if we compute with order $q = 4$ instead of $q = 2$. The mean errors are reduced to about 1/4 in both subdomains. Therefore, for oder $q = 4$ the solution is accurate to 4 K, and if we look at the mean error, it is accurate to 0.1 K.

Figure 5 shows the temperature T on the surface of the module and its error for the computation with consistency order $q = 4$. From the error picture, we can also see that the maximum errors occur only in few nodes.

For the solution of the resulting linear system of equations, we use the linear solver package LINSOL. The CPU time for LINSOL is about 99.5% of the total CPU time.

For the second problem, the leftmost five power chips have the same power dissipation of 250 W/Chip ($H = 15432.1\,\mathrm{W/cm}^3$, see (19)). Chip 6 has the same power dissipation, only in the area of the degraded array there is a heat generation density H with

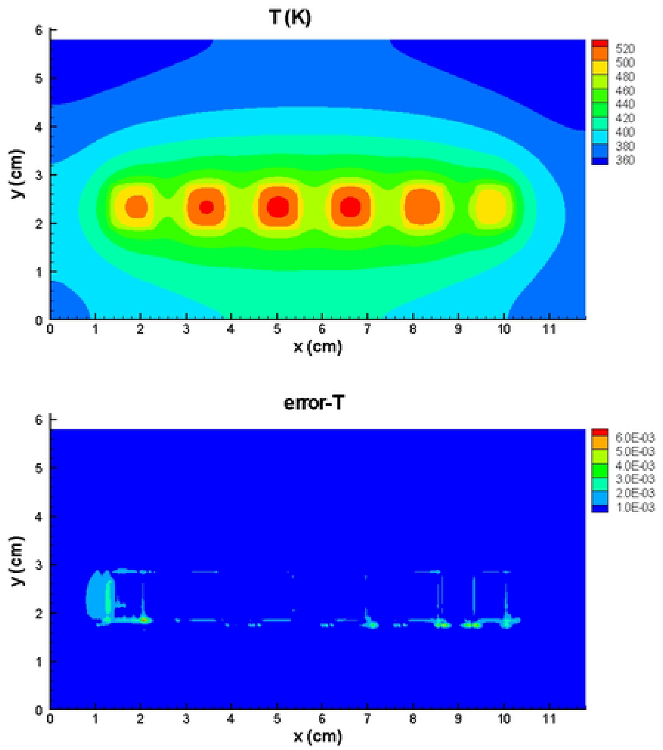

Fig. 5. Contour plot for the temperature T and its error on the top side of the module for $q = 4$ after 50 sec. ($H = 250$ W/Chip)

$$H = 250\,\text{W/chipvol.} + 50\,\text{W/degraded array vol.} = 265432.1\,\text{W/cm}^3\ . \quad (20)$$

In Table 8 we present the results of the second problem where the chips 1–5 have the same power dissipation of $H = 250$ W, but the 6th chip has a degraded array with an additional power dissipation of 50 W.

Table 8. Results of the second calculation with $H = 250$ W/Chip, 6th chip with degraded array

Order	Subd.	T_{max} (K)	rel. estim. error max.	rel. estim. error mean	CPU time (h)
$q = 2$	upper	706.3	$0.56 \cdot 10^{-1}$	$0.62 \cdot 10^{-3}$	6.6
	lower	632.7	$0.32 \cdot 10^{-1}$	$0.14 \cdot 10^{-3}$	
$q = 4$	upper	677.6	$0.15 \cdot 10^{-1}$	$0.13 \cdot 10^{-3}$	40.5
	lower	614.3	$0.73 \cdot 10^{-2}$	$0.26 \cdot 10^{-4}$	

For order $q = 2$, the maximum temperature, which is met in the centre of the degraded array of the 6^{th} chip, is 706.3 K after 50 sec. In the lower subdomain the maximum temperature is only 632.7 K. The maximum error in the upper domain is about 5.6%, in the lower subdomain it is only 3.2%, which means that the solution is accurate to 40 K. The mean errors are much smaller which means that the maximum errors occur only in few nodes. Considering the mean error, the solution is accurate to 0.5 K.

The maximum errors in the subdomains are reduced to 1/4 in both subdomains if we compute with order $q = 4$ instead of $q = 2$. The mean errors are reduced to about 1/5 in both subdomains. Therefore, for oder $q = 4$ the solution is accurate to 10 K, and if we look at the mean error, it is accurate to 0.1 K.

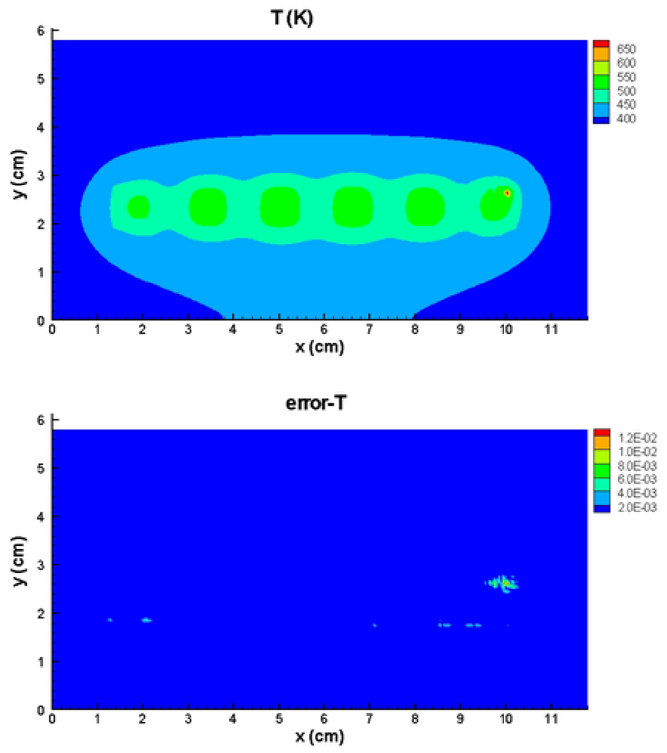

Fig. 6. Contour plot for the temperature T and its error on the top side of the module for $q = 4$ after 50 sec. (6^{th} chip with degraded array)

Figure 6 shows the temperature T on the surface of the module and its error for the computation with consistency order $q = 4$. Again, we see from the error pictures that the maximum errors occur only in few nodes.

Afterwards, we carried out some scalability computations. We repeated the computation with consistency order $q = 4$ and the degraded array on chip 6 on 64, 128, 256 and 512 processors. In Table 9, you can see the CPU time for master processor 1 for each computation.

Table 9. CPU times for the scalability test computations, grid with 312,750 nodes

No. of proc.	CPU time (h)
32	40.5
64	23.3
128	10.6
256	5.7
512	6.2

You see that the computation time is reduced by the factor 2 if we double the number of processors, at least up to 256 processors. For 512 processors the communication overhead strongly affects the computation time. As more than 99% of the computation time is comsumed by the linear solver LINSOL, there is still space for improvement.

This is not the usual way to examine the scalability of a code. So we also tried to double the number of nodes in the three space directions, and simultaneously compute with the eightfold number of processors. We tried to compute on 512 processors with a grid with 2,344,946 nodes, and wanted to compare the CPU time to that of the computation with 32 processors and 312,750 nodes. However, we must use LU preconditioning to solve the resulting linear system of equations, and then the factorization of the large sparse matrix is expected to consume the 32-fold computation time as we get the fourfold bandwidth of the matrix. So it is impossible to perform scalability tests this way.

We finally did measurements for the first time step with the grid with 2,344,946 nodes on 128, 256 and 512 processors. The CPU time for master processor 1 for the three computations are shown in Table 10.

Table 10. CPU times for the scalability test computations, grid with 2,344,946 nodes

No. of proc.	CPU time (min)
128	187.7
256	84.6
512	52.9

We see that the computation time is reduced by more than the expected factor 2 if we compare the computation on 256 processors to that on 128 processors, probably because of cache effects. For 512 processors, the communication overhead is the cause for the reduction factor of 1.6. For these computations, LINSOL consumed 99.99% of the CPU time.

6 Conclusion

We solved the PDE for the numerical simulation of the temperature in a lubrication gap of a high pressure Diesel injection pump. The error estimate showed the quality of the solution. Second, we solved the PDEs for the numerical simulation of the temperature in a power semiconductor module which is very challenging as it is a time-dependent problem in 3-D with two subdomains. This is the first time that problems of this type are solved with the knowledge of the error. This knowledge forces a very fine grid for a 1% error. So we need supercomputers for seemingly simple problems. The knowledge of the error reveals the true nature of the numerical complexity of these problems.

References

1. W. Schönauer, T. Adolph, FDEM: The Evolution and Application of the Finite Difference Element Method (FDEM) Program Package for the Solution of Partial Differential Equations, 2005, available at `www.rz.uni-karlsruhe.de/rz/docs/FDEM/Literatur/fdem.pdf`
2. T. Adolph, The Parallelization of the Mesh Refinement Algorithm in the Finite Difference Element Method, Doctoral Thesis, 2005, available at `www.rz.uni-karlsruhe.de/rz/docs/FDEM/Literatur/par_mra_fdem.pdf`
3. LINSOL, see `www.rz.uni-karlsruhe.de/rd/linsol.php`
4. T. Adolph, W. Schönauer, The Application of a Black-Box Solver with Error Estimate to Different Systems of PDEs, In: High Performance Computing in Science and Engineering '06 (ed. W. E. Nagel, W. Jäger, M. Resch), pp. 527–541, Transactions of the HLRS 2006, Springer
5. M. Petry, T. Adolph, W. Schönauer, The Snuffle Problem Petry for the Heat Conduction in a Thin Annulus, 2006, available at `www.rz.uni-karlsruhe.de/rz/docs/FDEM/Literatur/snuffle-petry.pdf`