Support Operator Technique for 3D Simulations of Dissipative Processes at High Performance Computers

Vladimir Gasilov, *Yuri Poveshenko*, *Irina Gasilova*, *Alexey Boldarev*, *Gennady Bagdasarov*, *Mikhail Yakobovskiy* and *Olga Olkhovskaya*

EASC 2015

THE UNIVERSITY of EDINBURGH

Motivation of using Support Operator Technique

Robust numerical algorithm with guaranteed quality for multiscale simulations

General 2D and 3D computational meshes (block-structured, unstructured, mortar etc.)

Conservative (fully conservative), monotonous (maximum principle), 2nd order difference schemes; self-adjoint, positive definite difference operator

Remarkable property: rotationally-invariant difference schemes

Support operator method for the second order differential equations

In the area O with the boundary ∂O let's consider common scalar-divergence boundary problem with, for example, Dirichlet boundary condition:

$$\begin{aligned} div\ \mathbf{X}_u &= f(\mathbf{r}), \\ \mathbf{X}_u &= K\ \nabla u \end{aligned} \qquad u\big|_{\partial O} = u_*(x) \tag{1}$$

Traditional approach: independent approximation of differential operators using the Gauss Theorem

$$\int_O \mathrm{div}\,\mathbf{X}\ dV = \int_{\partial O} \mathbf{X}\ d\mathbf{S}\ , \quad \int_O \mathrm{grad}\,\mathrm{u}\ dV = \int_{\partial O} \mathrm{u}\,d\mathbf{S}\,.$$

Our approach to a consistent approximation of differential operators: Support Operators Technique

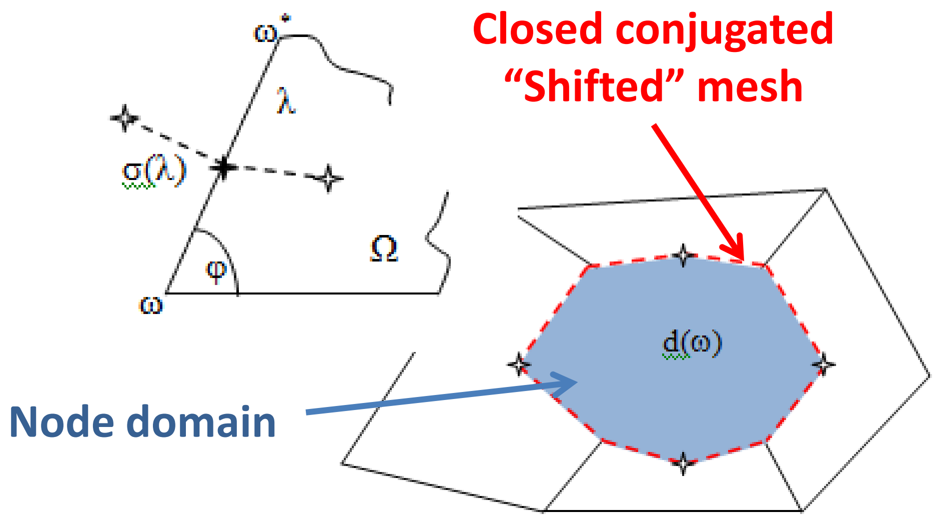
System (1) is considered along with (2). One operator is approximated directly; the other - in the way, that it satisfies the difference analogue of an integral identical equation :

$$\int_O (\mathbf{X}\ \nabla u)\ dv + \int_O u\ div\mathbf{X}\ dv = \int_{\partial O} u(\mathbf{X}, d\mathbf{s})\ . \tag{2}$$

The class of support operators schemes with ***grad*** as a support operator: metrical meshes

To construct the difference scheme it is needed to:

- introduce the difference grid in the computational domain
- define mesh functions on the grid, which approximate functions of the continuous argument.

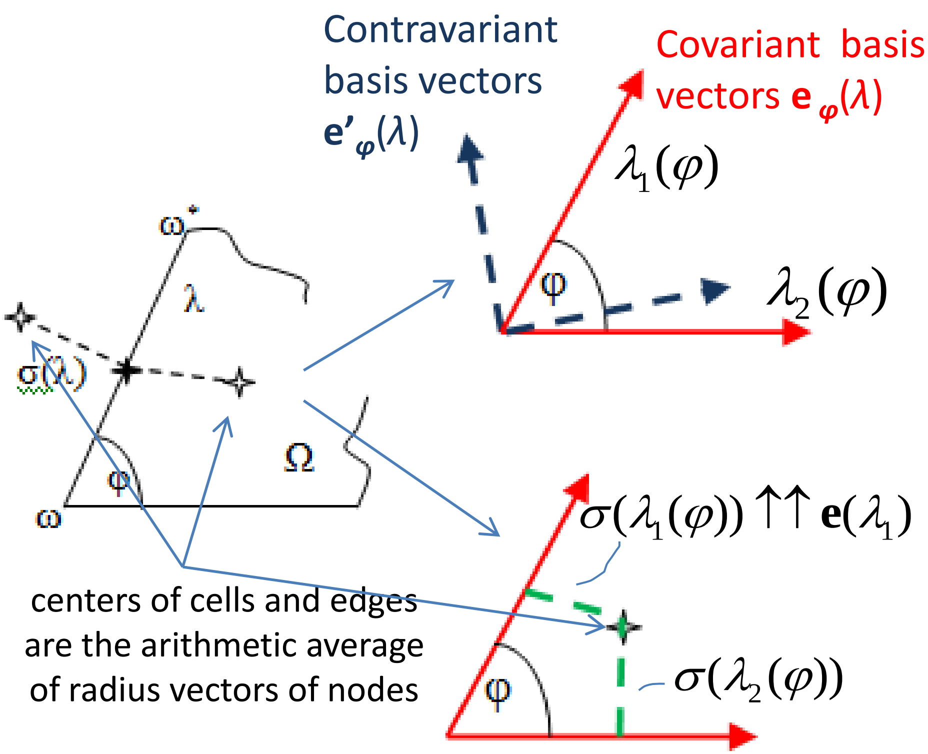


To the nodes of the grid (ω) we assign unknown mesh function u. In this case in the natural way operator ***grad*** is approximated.

Metrical meshes of support operators:

We cover the computational domain O with the difference grid of general type, which consists of:
nodes (ω),
formed by nodes cells-polygons (Ω),
bases (φ),
edges (λ)
linked to edges faces ($\sigma(\lambda)$) – boundaries of the balance node domains $d(\omega)$.
Closed around node ω surfaces $\sigma(\lambda(\omega))$ form node domains $d(\omega)$.

The class of support operators schemes with *grad* as a support operator: metrical operators



Bases are formed by the system of the initial (covariant) basis vectors , formed by edges.

Basis volume:

$$v_{\varphi} = \frac{1}{6}\left|\mathbf{e}(\lambda_1)\times\mathbf{e}(\lambda_2)\right| \quad \text{triangular cell}$$

$$v_{\varphi} = \frac{1}{4}\left|\mathbf{e}(\lambda_1)\times\mathbf{e}(\lambda_2)\right| \quad \text{quadrangular cell}$$

Contour, which links centers of the two adjacent by the edge cells or the cell with the boundary edge, represents an:

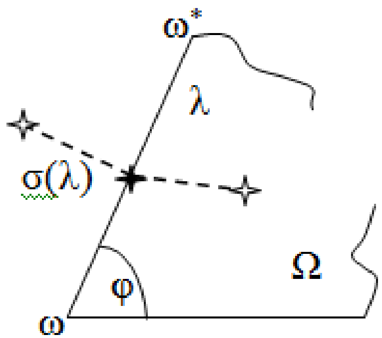
Edge surface: $\boldsymbol{\sigma}(\lambda) = \sum\limits_{\varphi(\lambda)} v_{\varphi}\mathbf{e}'_{\varphi}(\lambda)$

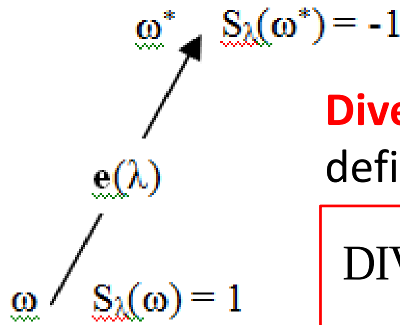
$\boldsymbol{\sigma}(\lambda) \uparrow\uparrow \mathbf{e}(\lambda)$

Metrical calibration of the difference mesh consists in the choice of the volumes of basis with the natural normalization condition $\sum\limits_{\varphi(\Omega)} V_{\varphi} = V_{\Omega}$.

Node volume: $\sum\limits_{\varphi(\omega)} v_{\varphi} = V_{\omega}$

The class of support operators schemes with *grad* as a support operator: approximation



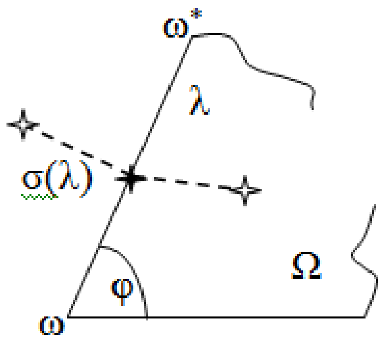


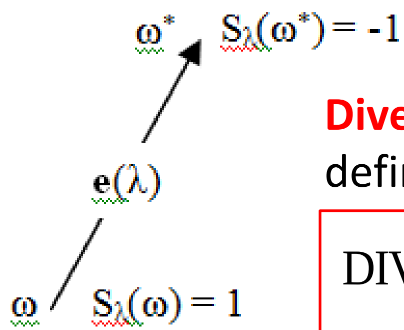
On the edges we choose the positive direction.

Divergence of the gradient field DIV: *(φ)*→*(ω)* we define by approximation of the Gauss Th. on *d(ω)*:

$$\boxed{\mathrm{DIV}\, X = \sum_{\lambda(\omega)} s_{\lambda}(\omega)\boldsymbol{\tau}_X(\lambda), \quad \boldsymbol{\tau}_X(\lambda) = \sum_{\varphi(\lambda)} \nu_{\varphi}\left(\mathbf{e}'_{\varphi}(\lambda),\, X_{\varphi}\right)}$$

The class of support operators schemes with *grad* as a support operator: approximation





On the edges we choose the positive direction.

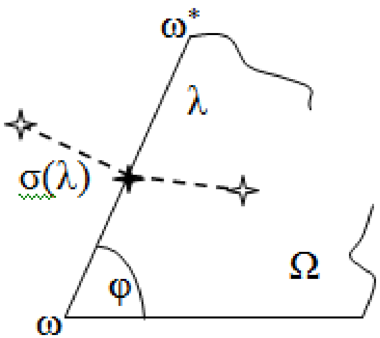
Divergence of the gradient field DIV: *(φ)→(ω)* we define by approximation of the Gauss Th. on *d(ω)*:

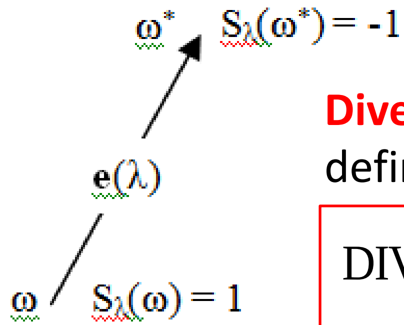
$$\mathrm{DIV}\, X = \sum_{\lambda(\omega)} s_\lambda(\omega)\boldsymbol{\tau}_X(\lambda), \quad \boldsymbol{\tau}_X(\lambda) = \sum_{\varphi(\lambda)} v_\varphi\left(\mathbf{e}'_\varphi(\lambda),\ X_\varphi\right)$$

Gradient vector field GRAD: *(ω)→(φ)* is given by its components in the bases:

$$\mathrm{GRAD}\, u = \sum_{\lambda(\varphi)} \Delta_\lambda u\ \mathbf{e}'_\varphi(\lambda), \quad \Delta_\lambda u = -\sum_{\omega(\lambda)} s_\lambda(\omega) u_\omega = u_{\omega^*} - u_\omega$$

The class of support operators schemes with *grad* as a support operator: approximation





On the edges we choose the positive direction.

Divergence of the gradient field DIV: *(φ)→(ω)* we define by approximation of the Gauss Th. on *d(ω)*:

$$\mathrm{DIV}\ X = \sum_{\lambda(\omega)} s_\lambda(\omega)\boldsymbol{\tau}_X(\lambda), \quad \boldsymbol{\tau}_X(\lambda) = \sum_{\varphi(\lambda)} v_\varphi \left(\mathbf{e}'_\varphi(\lambda),\ X_\varphi\right)$$

Gradient vector field GRAD: *(ω)→(φ)* is given by its components in the bases:

$$\mathrm{GRAD}\ u = \sum_{\lambda(\varphi)} \Delta_\lambda u\ \mathbf{e}'_\varphi(\lambda), \quad \Delta_\lambda u = -\sum_{\omega(\lambda)} s_\lambda(\omega) u_\omega = u_{\omega^*} - u_\omega$$

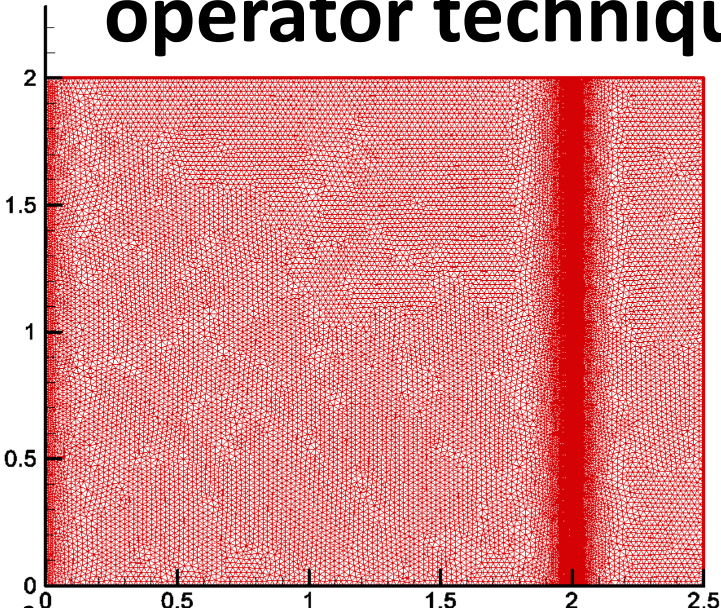
$(\ )_\Delta$ – approximation of the correspondent differential expressions, so we have:

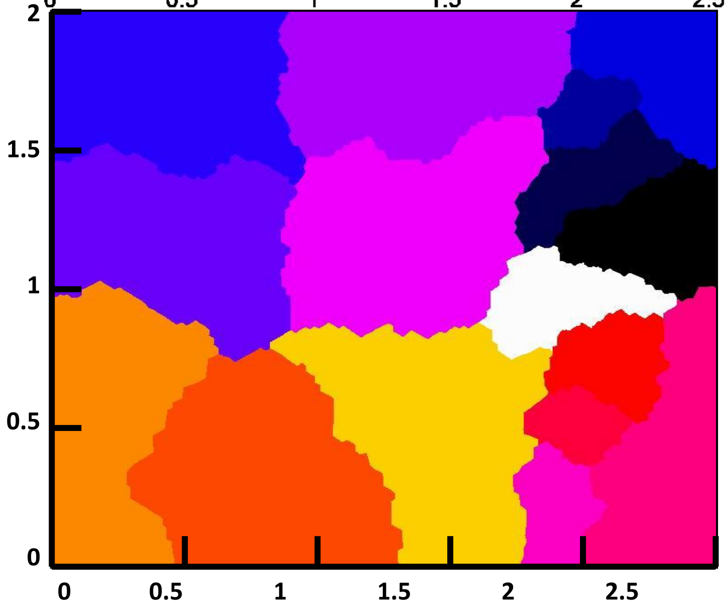
$$\left(\int_O (\mathbf{X}\,\nabla u)\ dv\right)_\nabla = -\left(\int_O u\ DIV\ \mathbf{X}\ dv - \int_{\partial O} u(\mathbf{X}, d\mathbf{s})\right)_\nabla = -\sum_\omega (u_\omega, DIV\ \mathbf{X}) = \sum_\varphi (\mathbf{X}_\varphi,\ \mathrm{GRAD}\ u)$$

Assuming in the bases *φ* under ***X****φ* vector field ***X****υφ* =*K* *φ*GRAD*υ* , we obtain **self-adjoint non-negative operator** -DIV ***X*** *υ* : *(ω)→(ω)* or -DIV *K* GRAD : *(ω)→(ω).*

This operator will be **strictly positive**, if at least in one boundary node of the closed difference mesh the Dirichlet boundary value problem is defined

Implicit heat diffusion solver based on support operator technique: parallel implementation



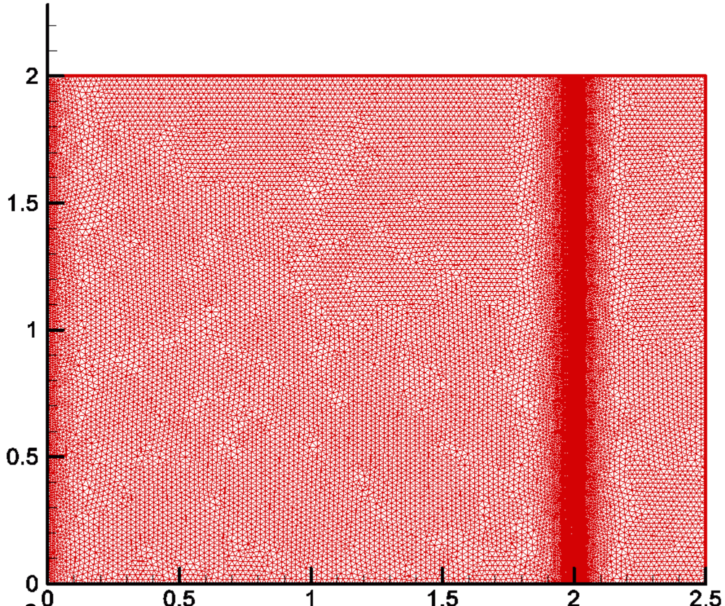


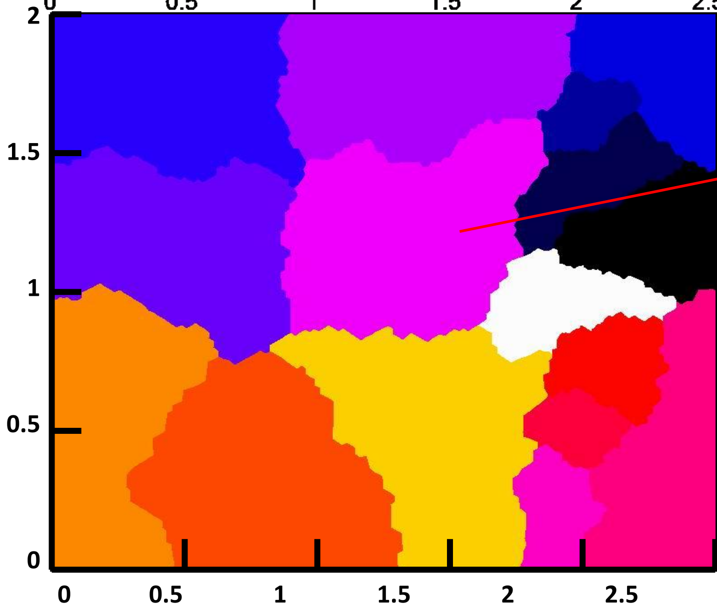
Solver is implemented within the MARPLE3D package; C++, MPI

Distributed algorithms are used in all stages of the problem solution:

- distributed mesh generation
- partitioning and repartitioning of meshes (ParMetis)
- parallel solution of the problem
- parallel analysis of the results (ParaView)

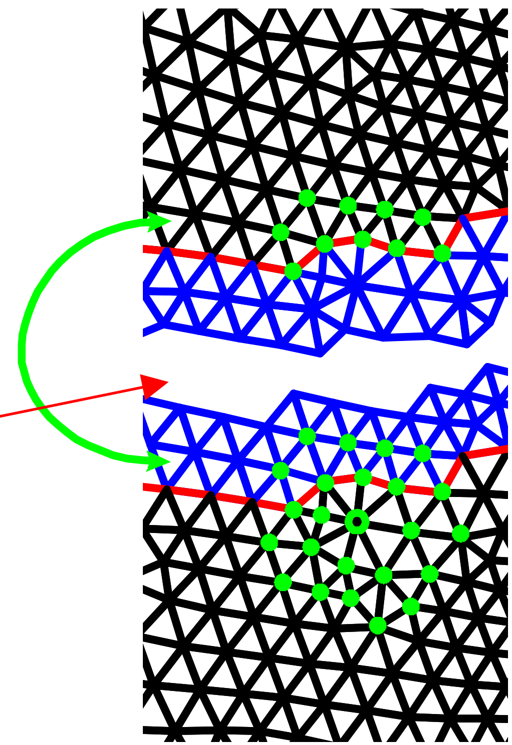
Domain decomposition for parallel computations





Implicit schemes lead to the **distributed systems of linear equations** (some equations involve the unknown values from the neighbouring subdomains). For the solution of these systems, the package **Aztec** is used.

Computations on the distributed mesh



“Fictive” mesh elements (margins)

Example of numerical simulations using support operator technique: propagation heat wave test

Let's consider heat diffusion equation in the following form:

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial s}\left(\kappa_0 \cdot T^{\alpha} \cdot \frac{\partial T}{\partial s}\right),$$

where T – unknown temperature, k_0 , α – free coefficients, s ≡ x|y|z. This equation with the following initial and boundary conditions:

$$T(s,\ 0) = \begin{cases} \left[\frac{\alpha D}{\kappa_0}(s_0 - s)\right]^{\frac{1}{\alpha}}, & s \leqslant s_0, \\ 0, & s > s_0, \end{cases}$$

$$T(0,\ t) = \left[\frac{\alpha D}{\kappa_0}(Dt + s_0)\right]^{\frac{1}{\alpha}}, \qquad t > 0,$$

has an analytical solution in the form of propagation with the constant velocity wave:

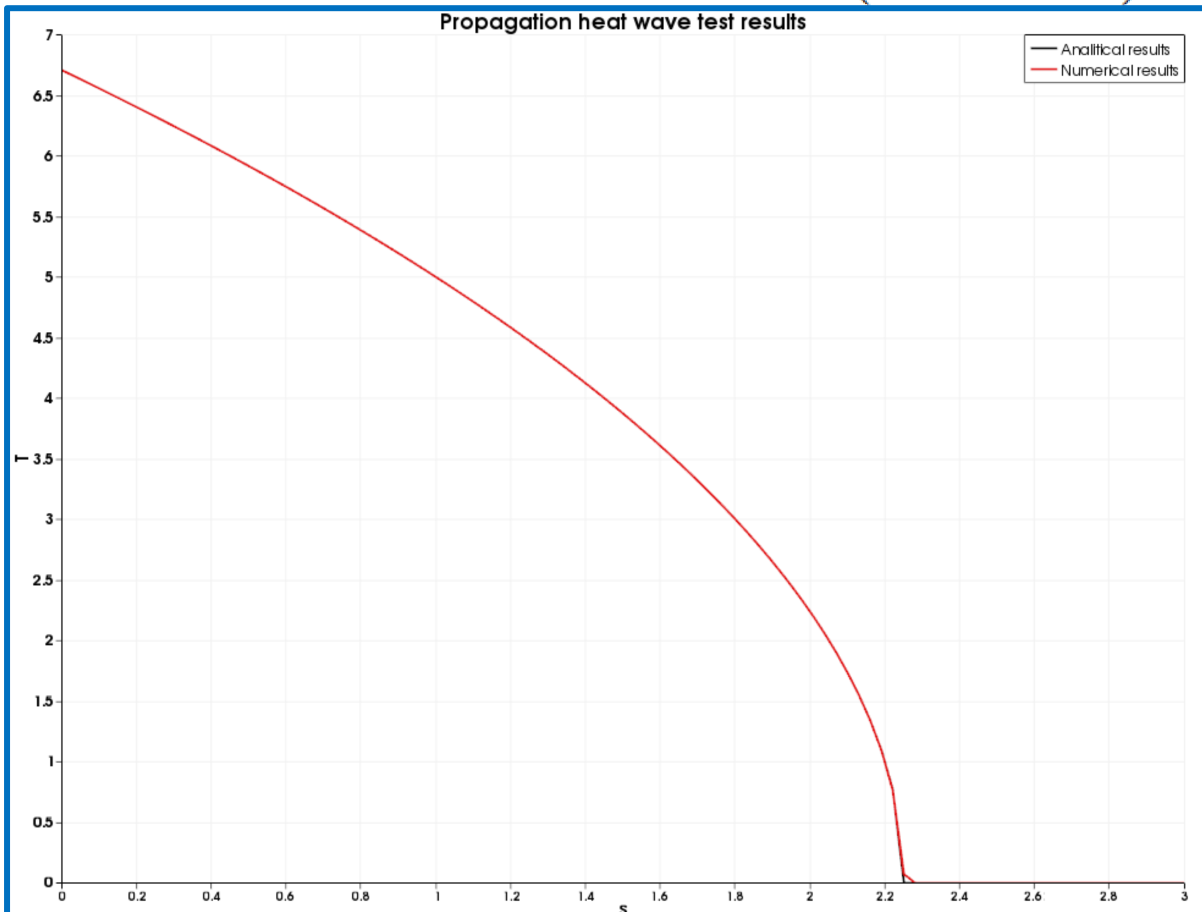
$$T(s,\ t) = \begin{cases} \left[\frac{\alpha D}{\kappa_0}(Dt + s_0 - s)\right]^{\frac{1}{\alpha}}, & s \leqslant s_0 + Dt, \\ 0, & s > s_0 + Dt, \end{cases}$$

where *D* – unknown temperature, s_0 – free parameter

Example of numerical simulations using support operator technique: propagation heat wave test

Let’s consider heat diffusion equation in the following form:

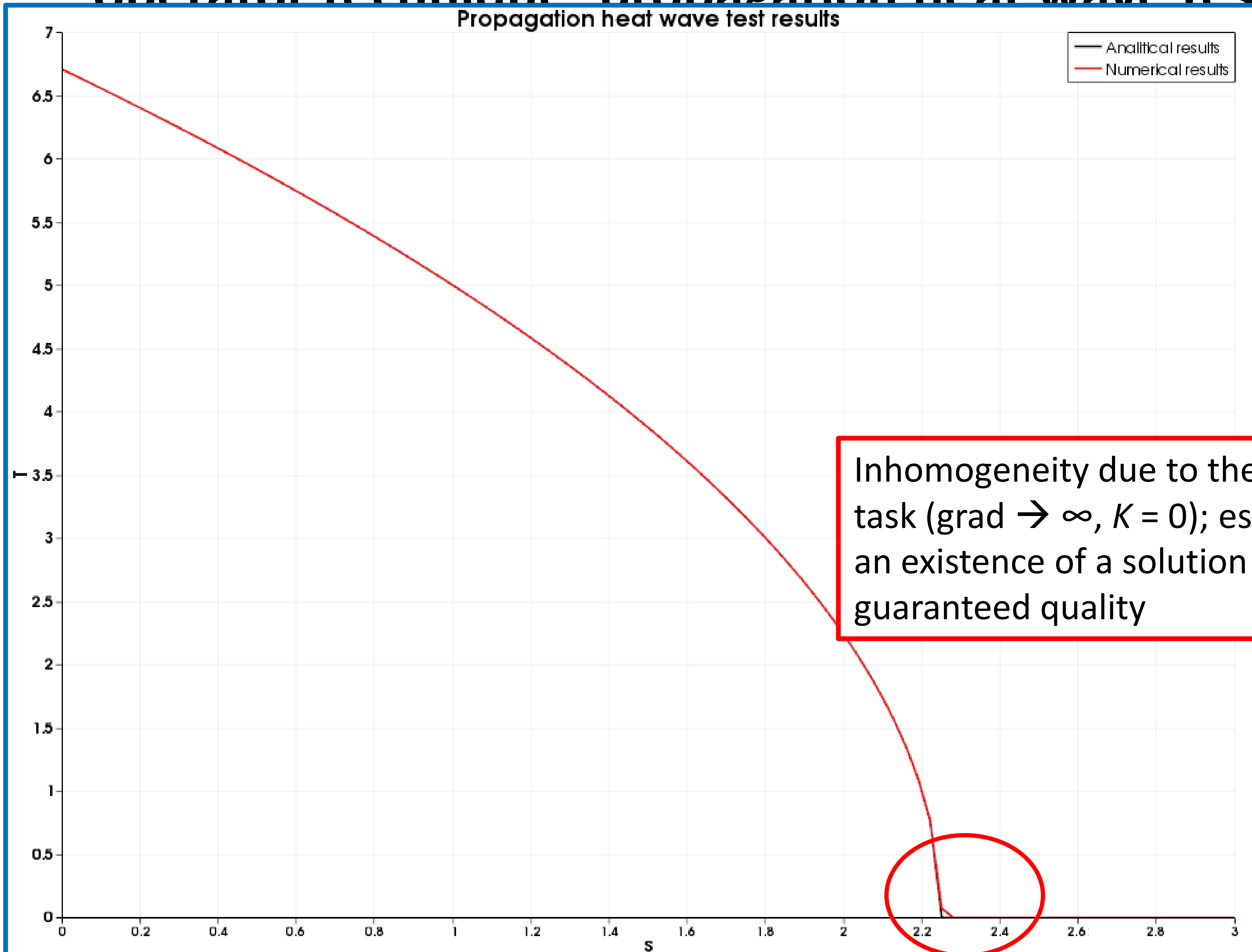
$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial s}\left(\kappa_0 \cdot T^{\alpha} \cdot \frac{\partial T}{\partial s}\right),$$



Simulations were implemented on 3D, 2D and 1D meshes with the typical size of *s* = 3

Test parameters:
$\alpha = 2.0,\ \kappa = 0.5,$
$s_0 = 0.5,\ D = 5.0;$
simulation time:
t from 0 to 0.4
with constant time step
$\tau = 2 \cdot 10^{-4}.$

Example of numerical simulations using support operator technique: propagation heat wave test

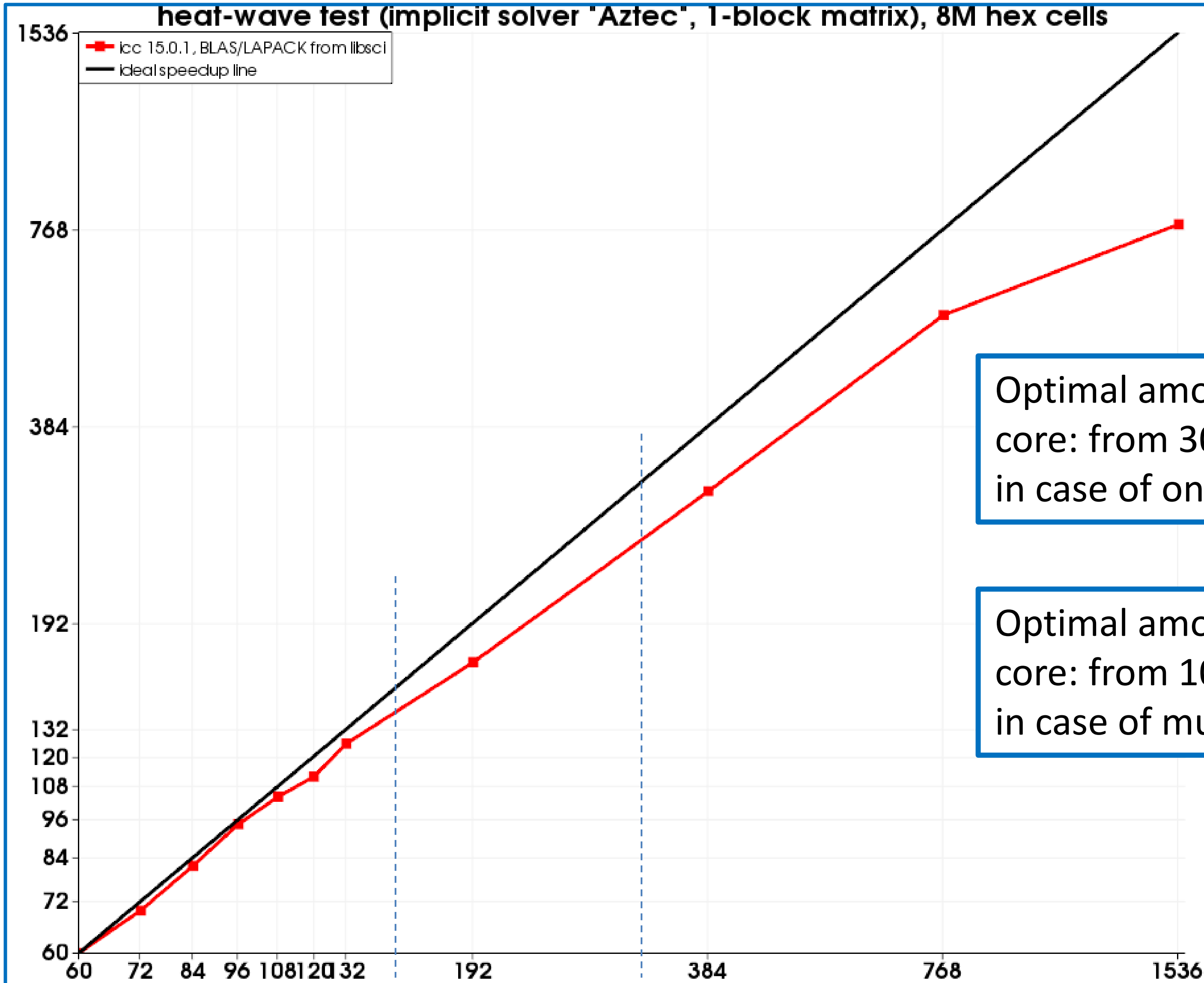


Inhomogeneity due to the physical task (grad → ∞, K = 0); essential an existence of a solution with a guaranteed quality

e
3D, 2D
with the
= 3

me step

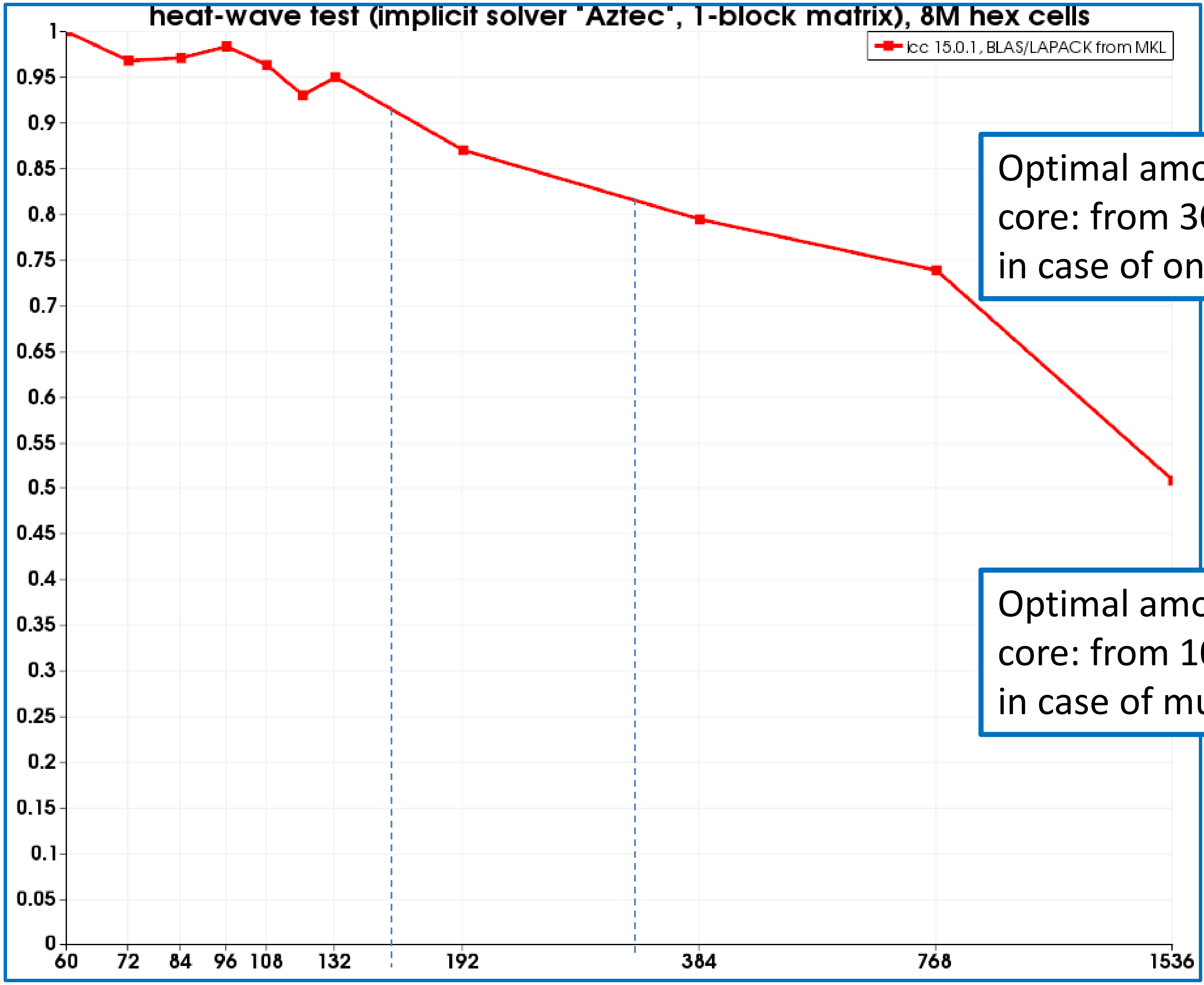
Results of scaling of developed software on NERSC's cluster: speed-up



Optimal amount of cells per core: from 30k to 50k
in case of one implicit solver

Optimal amount of cells per core: from 10k to 40k
in case of multiple solvers

Results of scaling of developed software on NERSC's cluster: efficiency



Optimal amount of cells per core: from 30k to 50k
in case of one implicit solver

Optimal amount of cells per core: from 10k to 40k
in case of multiple solvers

Conclusion

The numerical scheme based on support operator technique proves to be highly efficient for large parallel simulations of multiscale physical processes.

Future developments:

- *Implementation of support operators technique on mortar meshes.*
- *Developing of robust solvers for 3D linear and nonlinear elasticity problems within the MARPLE3D package.*
- *Developing our own GPU oriented solvers for solution of large sparse linear systems for effective hybrid computations*


Thank you for your attention!