

**CAVEAT-TR: A 2D hydrodynamic code with heat conduction
and radiation transport.**

I. Implementation of the SSI method for heat conduction

Mikhail M. Basko

*GSI, Darmstadt; on leave from ITEP, Moscow**

Joachim A. Maruhn and Anna Tauschwitz

Frankfurt University, Frankfurt

*Electronic address: basko@itep.ru

Contents

1. Introduction	5
2. Energy equation with thermal conduction and radiation transport	6
3. Lagrangian phase of the algorithm	8
3.1. The general 3D case	8
3.2. Reduction to the 2D case	9
4. General scheme of the SSI algorithm	11
5. SSI realization for heat conduction	13
5.1. General scheme	13
5.2. Formulae for explicit deposition rates W_T and total fluxes H_{ijm}	14
5.3. Flux limiting	15
5.4. Formulae for coefficients D_T	16
5.5. Formulae for the energy corrections δ_T	16
5.6. Weight coefficients for the energy correction	17
5.7. Time step control	18
5.8. Boundary conditions	19
1. Physical boundaries	19
2. Block interfaces	20
6. Test problems	23
6.1. Problem 1: one-dimensional non-linear heat wave	23
6.2. Problem 2: a spherical heat wave from an instantaneous point source	28
6.3. Problem 3: a planar shock wave with heat conduction	31
A. CAVEAT interpolation schemes	33
1. Vertex temperatures	33
a. 1D interpolation	33
b. 2D interpolation	34
2. Face-centered heat flux	35
3. Face-centered conduction coefficient	37
References	37

1. INTRODUCTION

The PHELIX laser at GSI (Darmstadt) is currently able to deliver about 500 joules of energy in 2–3 ns long pulses for plasma physics experiments. In future, the pulse energy is planned to be increased up to 3.6 kJ. A combination of intense laser and ion beams at GSI offers a unique possibility to study fundamental properties of matter at high energy densities. The PHELIX laser beam will be used to create hot plasmas with temperatures of up to 1 keV by direct target irradiation. Indirect-drive schemes, based on various forms of radiative hohlraums, hold out promise of well-controlled volumes of relatively dense quasi-uniform plasmas with temperatures $T \sim 100$ eV. These laser-generated plasmas will be used in a broad variety of experiments, some of them — like heavy ion stopping measurements in plasmas with $T \gtrsim 10$ –50 eV — quite unique. Also, as a powerful X-ray backlighter emitter, laser-created plasmas should find their place in diagnostics of various experiments, where matter is heated by intense ion beams. Investigation of hohlraum-created plasmas is of its own interest in the context of inertial confinement fusion research.

Preparation and interpretation of all sorts of experiments, involving high-temperature laser plasmas, usually requires sophisticated two-dimensional (2D) — if not three-dimensional — hydrodynamic simulations with a proper account of heat conduction and spectral radiation transport. However, development of such a 2D RH (radiative-hydrodynamics) code still provides a major challenge from the point of view of computational physics. And especially so, if one expects such a code to be reasonably universal, fast and accurate. Among a few 2D RH codes, existing presently in the world (like the well-known LASNEX code at Livermore), only the MULTI-2D code, developed by R. Ramis (Madrid, Spain), is freely available and could, in principle, be employed for laser-plasma simulations at GSI.

In numerical hydrodynamics there are two major competing approaches: one based on some type of artificial (pseudo-) viscosity (av-schemes), and the other, Godunov-like approach, based on some version of a Riemann solver [1]. From the physics point of view, a Godunov-like approach would always be preferable because it does not involve unphysical viscosity and is automatically well-suited for accurate description of large density and pressure jumps. Once higher-order low-diffusion Godunov-like schemes had been developed [2], they were demonstrated to be superior to traditional av-schemes [1] in terms of accuracy as well — at least to those of them which do not use specially developed techniques [3].

The MULTI-2D code by R. Ramis is based on a rather simple version of an av-scheme for an unstructured triangular mesh [4]. As a consequence, its capabilities in modelling complex two-dimensional flows (even in its later versions with a mesh adaptivity option) — like, for example, high-convergence implosions — appear rather limited. Also, radiation transport in the distributed version of MULTI-2D is available only in a single-frequency-group approximation. An additional obstacle for any attempt to modify or upgrade the MULTI-2D code is the need to master the author's original high-level programming language, in which the source code is written.

At the same time, very positive experience has been accumulated at GSI [5, 6] with the 2D purely hydrodynamical code CAVEAT, originally developed at the Los Alamos National Laboratory [7] and written in the standard FORTRAN-77. CAVEAT is based on a second-order Godunov-like scheme for a structured quadrilateral grid, and fully exploits the advantages of the arbitrary Lagrangian-Eulerian (ALE) technique. Its numerical scheme produces just the right amount of numerical diffusion, so that, for example, in the Rayleigh-

Taylor unstable situations the shortest (on the scale of a single grid cell) wavelengths are conveniently suppressed, whereas at wavelengths of 8–10 grid cells the instability is quite accurately reproduced [5].

An important advantage of CAVEAT is that the scheme is able to maintain very accurately the inherent symmetry of the problem (due to the structured character of the mesh and moderate amount of numerical diffusion) and, for example, follow symmetric implosions down to the radial convergence factors of $\simeq 100$ [6]. The code is well suited for treating multi-material problems with a broad selection of different equations of state; a special attention is paid to accurate tracing of material interfaces. Another major advantage of CAVEAT is a possibility to handle topologically complex situations, where the computational domain is comprised of many mutually bordering topologically quadrilateral blocks, each having a locally structured mesh. Building up on this positive experience, a decision was made by the present authors to undertake a challenging project of upgrading the original 2D CAVEAT code to a version which includes both the *thermal conduction* and the *spectral radiation transport*. The new version has been named **CAVEAT-TR**.

The original CAVEAT scheme is based on cell-centered values for all principal dynamic variables, one of which is the total (internal + kinetic) fluid energy density E . For thermal conduction and radiation transport, a more natural principal variable would be matter temperature T , for which vertex-centered values are needed. The difficulties associated with combining these two approaches are extensively discussed in Ref. [8]. In this part I of the report on CAVEAT-TR we describe implementation of the symmetrical semi-implicit (SSI) method [9] for treating thermal conduction. After introduction of two complementary criteria for time-step control, the SSI method was found to work quite accurately and efficiently within the general CAVEAT scheme. The adopted reinterpolation scheme from the cell-centered to vertex-centered temperature values is demonstrated to provide more than sufficient accuracy in describing the propagation of non-linear heat fronts.

2. ENERGY EQUATION WITH THERMAL CONDUCTION AND RADIATION TRANSPORT

Heat conduction and radiation transport (the latter — in the simplest approximation of instantaneous energy redistribution) appear only in the energy equation

$$\frac{\partial(\rho E)}{\partial t} + \text{div} [(\rho E + p)\vec{u}] = - \text{div} (\vec{h} + \vec{h}_r) + Q \quad (2.1)$$

of the full system of the hydrodynamic equations. Here

$$\begin{aligned}
\rho = \rho(t, \vec{x}) & \quad \text{is the fluid density [g cm}^{-3}\text{]}, \\
E = e + \frac{u^2}{2} & \quad \text{is the mass-specific total energy [erg g}^{-1}\text{]}, \\
e = e(t, \vec{x}) & \quad \text{is the mass-specific internal energy [erg g}^{-1}\text{]}, \\
\vec{u} = \vec{u}(t, \vec{x}) & \quad \text{is the fluid velocity [cm s}^{-1}\text{]}, \\
p = p(\rho, e) & \quad \text{is the fluid pressure [erg cm}^{-3}\text{]}, \\
Q = Q(t, \vec{x}) = \rho q & \quad \text{is the volume-specific external heating rate [erg cm}^{-3} \text{s}^{-1}\text{]}, \\
q = q(t, \vec{x}) & \quad \text{is the mass-specific external heating rate [erg g}^{-1} \text{s}^{-1}\text{]}, \\
T = T(\rho, e) & \quad \text{is the fluid temperature [eV]}, \\
\vec{h} = -\kappa \nabla T & \quad \text{is the area-specific heat flux [erg cm}^{-2} \text{s}^{-1}\text{]}, \\
\kappa = \kappa(\rho, T) & \quad \text{is the heat conduction coefficient [erg cm}^{-1} \text{s}^{-1} \text{eV}^{-1}\text{]}, \\
\vec{h}_r = \int_0^\infty d\nu \int_{4\pi} \vec{\Omega} I d\vec{\Omega} & \quad \text{is the area-specific radiation energy flux [erg cm}^{-2} \text{s}^{-1}\text{]}, \\
I = I(\vec{x}, \nu, \vec{\Omega}) & \quad \text{is the radiation intensity [erg cm}^{-2} \text{s}^{-1} \text{ster}^{-1} \text{eV}^{-1}\text{]}, \\
\nu & \quad \text{is the photon frequency [eV]}, \\
\vec{\Omega} & \quad \text{is the unit vector in the direction of photon propagation,} \\
d\vec{\Omega} & \quad \text{is the solid angle element.}
\end{aligned} \tag{2.2}$$

At each time t the radiation intensity $I(\vec{x}, \nu, \vec{\Omega})$ is calculated by solving the time-independent transport equation

$$\vec{\Omega} \cdot \nabla I = k_\nu (I_{pl} - I), \tag{2.3}$$

where

$$\begin{aligned}
k_\nu = k_\nu(\nu, \rho, T) & \quad \text{is the radiation absorption coefficient [cm}^{-1}\text{]}, \\
I_{pl} = 5.0404 \times 10^{10} \frac{\nu^3}{\exp(\nu/T) - 1} & \quad \text{is the Planckian intensity [erg cm}^{-2} \text{s}^{-1} \text{ster}^{-1} \text{eV}^{-1}\text{]}.
\end{aligned} \tag{2.4}$$

Note that k_ν is the full — i.e. corrected for stimulated emission — absorption coefficient. The simplest version (2.3) of the transport equation is written under the assumptions that (i) at each time a negligible amount of energy resides in the radiation field, and (ii) no momentum is transferred by radiation, i.e. radiation pressure can be ignored. This approximation may be called the approximation of *instantaneous energy redistribution* (IRD), and is in this sense analogous to heat conduction: in the optically thick case it is equivalent to adding the radiative heat conduction coefficient $\kappa_r = \frac{4}{3} a c l_R T^3$ to the molecular (electron) heat conduction coefficient κ ; here a is a constant in the expression aT^4 [erg cm⁻³] for the equilibrium radiative energy density, c is the speed of light, and $l_R = l_R(\rho, T)$ is the Rosseland mean free path of photons.

3. LAGRANGIAN PHASE OF THE ALGORITHM

3.1. The general 3D case

In the CAVEAT code all the energy deposition and redistribution is accomplished during the Lagrangian phase (L-step) of the computational cycle; a computational cycle is a single time step, during which all the principal variables are advanced in time from t to $t + \Delta t$. Below the quantities in a cell (i, j) at the end of the Lagrangian phase are marked with a bar, like \bar{e}_{ij} , \bar{E}_{ij} , \bar{T}_{ij} ...

In the Lagrangian phase the mass-specific total energy E is advanced according to the integral form of the energy equation (2.1)

$$\frac{d}{dt} \int_{V_{ij}} \rho E dV = - \int_{S_{ij}} p \vec{u} \cdot \vec{n} dS - \int_{S_{ij}} (\vec{h} + \vec{h}_r) \cdot \vec{n} dS + \int_{V_{ij}} \rho q dV, \quad (3.1)$$

where V_{ij} is the Lagrangian volume of the mesh cell (i, j) , S_{ij} is its surface area, and \vec{n} is the *outward* unit normal to the cell surface. Note that the cell mass

$$M_{ij} = \int_{V_{ij}} \rho dV \quad (3.2)$$

is conserved in the L-step, i.e. $dM_{ij}/dt = 0$. By definition, the following relationships are always fulfilled

$$E_{ij} = \frac{1}{M_{ij}} \int_{V_{ij}} \rho E dV, \quad q_{ij} = \frac{1}{M_{ij}} \int_{V_{ij}} \rho q dV. \quad (3.3)$$

The density ρ_{ij} in cell (i, j) is calculated from the relationship

$$\rho_{ij} = \frac{M_{ij}}{V_{ij}}. \quad (3.4)$$

The mass-specific total energy E_{ij} of cell (i, j) is advanced in the L-step according to the following finite-difference approximation to Eq. (3.1)

$$\bar{E}_{ij} = E_{ij} + \Delta t \left(\frac{W_{p,ij} + W_{T,ij} + W_{r,ij}}{M_{ij}} + q_{ij} \right), \quad (3.5)$$

where the corresponding energy deposition powers (in [erg s⁻¹]) in cell (i, j) are given by

$$W_{p,ij} = - \int_{S_{ij}} p \vec{u} \cdot \vec{n} dS, \quad W_{T,ij} = - \int_{S_{ij}} \vec{h} \cdot \vec{n} dS, \quad W_{r,ij} = - \int_{S_{ij}} \vec{h}_r \cdot \vec{n} dS. \quad (3.6)$$

The cell volume V_{ij} is advanced in the L-step according to the equation

$$\frac{dV_{ij}}{dt} = \int_{V_{ij}} \text{div } \vec{u} dV = \int_{S_{ij}} \vec{u} \cdot \vec{n} dS. \quad (3.7)$$

Correspondence with the code variables:

$\rho_{ij} =$	RHO(I,J)	intensive, mass density;
$p_{ij} =$	PR(I,J)	intensive, pressure;
$V_{ij} =$	VOL(I,J)	extensive, cell volume;
$M_{ij} =$	CM(I,J)	extensive, cell mass; conserved in the L-step;
$e_{ij} =$	SIE(I,J)	intensive, mass-specific internal energy;
$\bar{e}_{ij} =$	SIEL(I,J)	intensive;
$E_{ij} =$	TE(I,J)	intensive;
$\bar{E}_{ij} =$	TEL(I,J)	intensive;
$q_{ij} =$	QDEPO(I,J)	intensive, mass-specific heating rate;

3.2. Reduction to the 2D case

Two-dimensional flows are described by using two different versions of coordinate systems: (i) Cartesian, when the internal CAVEAT coordinates are $(x_1, x_2) = (x, y)$, and (ii) cylindrical, when the internal coordinates are either $(x_1, x_2) = (r, z)$, or $(x_1, x_2) = (z, r)$; here r is the cylindrical radius. These two different mesh geometries are combined by introducing a generalized radius

$$R = \begin{cases} 1, & (x_1, x_2) = (x, y), \text{ Cartesian, } \text{IRADIAL}=0, \\ x_1, & (x_1, x_2) = (r, z), \text{ cylindrical 1, } \text{IRADIAL}=1, \\ x_2, & (x_1, x_2) = (z, r), \text{ cylindrical 2, } \text{IRADIAL}=2. \end{cases} \quad (3.8)$$

Then, the 3D volume V_{ij} in 2D problems becomes either a *volume per unit length* along the z-axis in the Cartesian case, or a *volume per steradian of the azimuth angle* in the axially symmetric case. To get the full 3D volume in the axially symmetric case, one has to multiply the 2D volume V_{ij} by 2π . The same pertains to the surface areas S_{ij} , and to all extensive physical quantities.

Accordingly, all the volume and surface integrals are transformed as follows. The integral of any scalar intensive quantity q over the cell volume is cast in a form

$$\int_{V_{ij}} q dV = \int_{A_{ij}} q R dx_1 dx_2, \quad (3.9)$$

where A_{ij} is the surface area of the mesh cell (i, j) on the x_1, x_2 -plane; remind that the internal coordinates x_1, x_2 are treated as orthogonal Cartesian.

A corresponding surface integral is then given by

$$\int_{S_{ij}} q dS = \int_{L_{ij}} q R d\lambda, \quad (3.10)$$

where L_{ij} is the perimeter of the mesh cell (i, j) on the x_1, x_2 -plane, and $d\lambda$ is the length element along this perimeter. For quadrangular cells with straight sides the integral (3.10)

is approximated as a sum over its four sides (faces)

$$\int_{L_{ij}} q R d\lambda = \sum_{\alpha=1}^4 q_{f,\alpha} \Lambda_{\alpha}, \quad (3.11)$$

where $q_{f,\alpha}$ is the face-centered value of q , and

$$\Lambda_{\alpha} = \int_{\text{face } \alpha} R d\lambda. \quad (3.12)$$

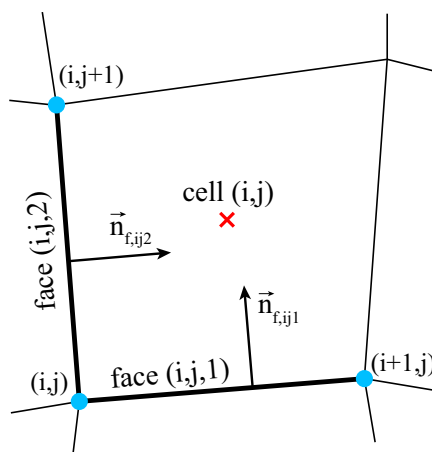


FIG. 3.1: Numbering of vertices, cells and faces in CAVEAT.

For straight-line faces the integral in (3.12) is calculated exactly. In the CAVEAT nomenclature the faces are numbered after the vertices from which they originate; for each vertex (i, j) there are two faces ascribed to it; face $(i, j, 1)$ connects the vertex (i, j) to $(i + 1, j)$; face $(i, j, 2)$ connects the vertex (i, j) to $(i, j + 1)$ (see Fig. 3.1). On each face (i, j, m) a unit normal vector $\vec{n}_{f,ijm}$ is defined, which points in the positive direction of the alternative coordinate x_{3-m} ; here $m = 1, 2$. This normal vector should not be mixed (!) with the outward normal \vec{n} to the cell surface, used in the formulae of the previous section. The quantity Λ_{ijm} is called the *area of the face* (i, j, m) ; its values are given by

$$\begin{aligned} \Lambda_{ij1} &= \frac{1}{2}(R_{ij} + R_{i+1,j}) |\vec{x}_{i+1,j} - \vec{x}_{ij}|, \\ \Lambda_{ij2} &= \frac{1}{2}(R_{ij} + R_{i,j+1}) |\vec{x}_{i,j+1} - \vec{x}_{ij}|; \end{aligned} \quad (3.13)$$

here $\vec{x} = \{x_1; x_2\}$ is the vector with coordinates x_1, x_2 , and \vec{x}_{ij} represents coordinates of the mesh vertex (ij) (see Fig. 3.1).

Correspondence with the code variables:

$x_{1,ij} =$	XV(I,J,1)	x_1 -coordinate of vertex (i, j) ;
$x_{2,ij} =$	XV(I,J,2)	x_2 -coordinate of vertex (i, j) ;
$n_{f,ijm,1} =$	FN(I,J,M,1)	x_1 -component of the unite normal vector $\vec{n}_{f,ijm}$; calculated in subroutine GEOM;
$n_{f,ijm,2} =$	FN(I,J,M,2)	x_2 -component of the unite normal vector $\vec{n}_{f,ijm}$; calculated in subroutine GEOM;
$\Lambda_{ijm} =$	FA(I,J,M) (> 0)	area of face (i, j, m) ; calculated in subroutine GEOM;

4. GENERAL SCHEME OF THE SSI ALGORITHM

When applying the SSI method to Eq. (3.5), we split the terms on its right-hand side into two groups: (i) the $p dV$ -work, represented by the $\Delta t W_{p,ij}/M_{ij}$ term, is treated explicitly (i.e. taken over unaltered from the original CAVEAT version) and remains untouched by the SSI scheme; (ii) the remaining $\Delta t W_{T,ij}/M_{ij}$, $\Delta t W_{r,ij}/M_{ij}$, and $\Delta t q_{ij}$ terms are combined into a joint SSI step. As discussed below, splitting into separate SSI steps with respect to heat conduction and radiative transport might seriously undermine the code's capability to treat laser-heating problems.

Then, the L-step for the energy equation is accomplished according to the following finite-difference representation

$$\bar{E}_{ij} = E_{ij} + \frac{\Delta t}{M_{ij}} W_{p,ij} + \frac{\Delta t}{M_{ij}} \left(\tilde{W}_{T,ij} + \tilde{W}_{r,ij} \right) + q_{ij} \Delta t + \frac{\delta_{T,ij} + \delta_{r,ij}}{M_{ij}}. \quad (4.1)$$

The cell deposition powers $W_{T,ij}$, $W_{r,ij}$ [erg s⁻¹] are defined in Eq. (3.6). Tilde above these quantities means that they are evaluated by using not the “old” (i.e. assigned before the start of the new computational cycle) temperatures T_{ij} but the SSI-advanced “new” temperatures \tilde{T}_{ij} (for more details see the next section); note that the SSI-advanced temperatures \tilde{T}_{ij} are not the same as the L-step-advanced temperatures \bar{T}_{ij} . The values $W_{T,ij}$, $W_{r,ij}$ without a tilde are calculated by using the old temperatures T_{ij} ; it is these values that would have been used under the explicit treatment of heat conduction and radiation transport. The terms $\delta_{T,ij}$, $\delta_{r,ij}$ [erg] in Eq. (4.1) are the energy corrections from the previous time step, originating from the energy disbalance inherent in the SSI method [9]. These quantities do not participate in calculation of the SSI-advanced temperatures \tilde{T}_{ij} and deposition powers $\tilde{W}_{T(r),ij}$.

When adding the amount of energy $\delta_{T(r),ij}$ to cell (i, j) , it should be kept in mind that $\delta_{T(r),ij}$ was calculated in the previous hydro cycle. Since due to mesh rezoning the cell mass M_{ij} may change from cycle to cycle, the “old” energy correction, imposed on a “new” mass M_{ij} may, in principal, lead to a temperature overshoot and potential instability. Nevertheless, we choose this method, based on passing to a new cycle the absolute energy correction $\delta_{T(r),ij}$ rather than the mass-specific quantity $\delta_{T(r),ij}/M_{ij}$, because it is globally conservative, and because it is difficult to propose a reasonable alternative for a purely Eulerian mesh.

Probably, the energy corrections $\delta_{T,ij}$ and $\delta_{r,ij}$ should be also subjected to the remapping procedure at the rezoning phase!

Now, having split the physical processes into those participated and not participating in the SSI step, we have the following equation for calculating the SSI-advanced temperatures \tilde{T}_{ij} and specific internal energies \tilde{e}_{ij} :

$$\tilde{e}_{ij} - e_{ij} = \frac{\Delta t}{M_{ij}} \left(\tilde{W}_{T,ij} + \tilde{W}_{r,ij} + q_{ij} M_{ij} \right) + \frac{\delta_{T,ij} + \delta_{r,ij}}{M_{ij}}. \quad (4.2)$$

Because the SSI step does not include the hydro motion, we use the specific internal energies e_{ij} instead of E_{ij} , and we can relate the change in e to the change in T by invoking the heat capacity c_V (per unit mass) at constant volume,

$$\tilde{e}_{ij} - e_{ij} = c_{V,ij} \tau_{ij}, \quad (4.3)$$

where

$$\tau_{ij} = \tilde{T}_{ij} - T_{ij} \quad (4.4)$$

is the principal unknown quantity to be determined at the SSI step. The heat capacity c_V is calculated from the old (i.e. available at the start of the cycle) values of e, ρ, T .

After an appropriate linearization (required, in particular, for the radiative transport), general expressions for $\tilde{W}_{T(r),ij}$ can be cast in the form

$$\tilde{W}_{T,ij} = -D_{T,ij} \tau_{ij} + W_{T,ij}, \quad \tilde{W}_{r,ij} = -D_{r,ij} \tau_{ij} + W_{r,ij}, \quad (4.5)$$

where the coefficients $D_{T,ij}$ are determined by mesh geometry and the conduction coefficients κ_{ij} , $D_{r,ij}$ — by mesh geometry, opacities $k_{\nu,ij}$, old temperatures T_{ij} and densities ρ_{ij} . Note that by their physical meaning the coefficients $D_{T(r),ij}$ must be non-negative because, when a cell (i, j) is overheated and τ_{ij} is large, both heat conduction and radiative transport should lead to the cooling of this same cell (i, j) .

Having substituted Eqs. (4.5) and (4.3) into Eq. (4.2), we obtain

$$\tau_{ij} = \frac{(W_{T,ij} + W_{r,ij} + q_{ij} M_{ij}) \Delta t + \delta_{T,ij} + \delta_{r,ij}}{c_{V,ij} M_{ij} + \Delta t (D_{T,ij} + D_{r,ij})}, \quad (4.6)$$

and the final equation for the L-step in the specific total energy

$$\bar{E}_{ij} = E_{ij} + \frac{\Delta t}{M_{ij}} W_{p,ij} + c_{V,ij} \tau_{ij}. \quad (4.7)$$

Obviously, for $D_{T,ij} = D_{r,ij} = 0$ we recover the “underlying” explicit scheme for heat conduction and radiation transport.

Equation (4.6) clearly illustrates the advantages of combining conduction, radiation and external heating into a single SSI step (i.e. not splitting with respect to each individual process). If, for example, laser heating occurs in a thin layer, then this layer rapidly becomes very hot with large values of either $D_{T,ij}$ or $D_{r,ij}$ (or both); then, once $\Delta t (D_{T,ij} + D_{r,ij}) > c_{V,ij} M_{ij}$, Eq. (4.6) naturally prevents the heated cells from being “overheated” — which would occur with a separate and explicit treatment of the heating source q_{ij} .

Correspondence with the code variables:

$\Delta t =$	DTHYDRO	time step;
$T_{ij} =$	TEMP(I,J)	intensive, temperature;
$\kappa_{ij} =$	TCOND(I,J)	intensive, conduction coefficient;
$c_{V,ij} =$	CV(I,J)	intensive, mass-specific heat capacity;
$\tilde{e}_{ij} - e_{ij} =$	DTESSI(I,J)	intensive, mass-specific internal energy increment;
$\delta_{T,ij} + \delta_{r,ij} =$	ECORSSI(I,J)	extensive, energy correction to cell (i, j) for the next cycle;

5. SSI REALIZATION FOR HEAT CONDUCTION

5.1. General scheme

As explained in section 4, implementation of the SSI algorithm for heat conduction requires calculation of three cell-centered quantities: $W_{T,ij}$, $D_{T,ij}$ and $\bar{\delta}_{T,ij}$. Bar over $\delta_{T,ij}$ means that this quantity will only be used in the next cycle. The deposition power $W_{T,ij}$ for cell (i, j) can be written as

$$\begin{aligned} W_{T,ij} &= H_{ij1} + H_{ij2} - H_{i,j+1,1} - H_{i+1,j,2}, \\ \tilde{W}_{T,ij} &= \tilde{H}_{ij1} + \tilde{H}_{ij2} - \tilde{H}_{i,j+1,1} - \tilde{H}_{i+1,j,2}, \end{aligned} \quad (5.1)$$

where

$$H_{ijm} = + \int_{\text{face } (i,j,m)} (\kappa \nabla T \cdot \vec{n}) R d\lambda = - \int_{\text{face } (i,j,m)} (\kappa \nabla T \cdot \vec{n}_{f,ijm}) R d\lambda, \quad m = 1, 2, \quad (5.2)$$

is the total (in [erg s⁻¹]) conductive energy flux across the face (i, j, m) (see Fig. 5.1). In a finite difference version of Eq. (5.2) H_{ijm} is a linear function of the temperatures T_{ij} , $T_{i\pm 1,j}$, $T_{i,j\pm 1}$, $T_{i\pm 1,j\pm 1}$, associated with the centers of the cell (i, j) and the neighboring cells $(i \pm 1, j)$, $(i, j \pm 1)$, $(i \pm 1, j \pm 1)$.

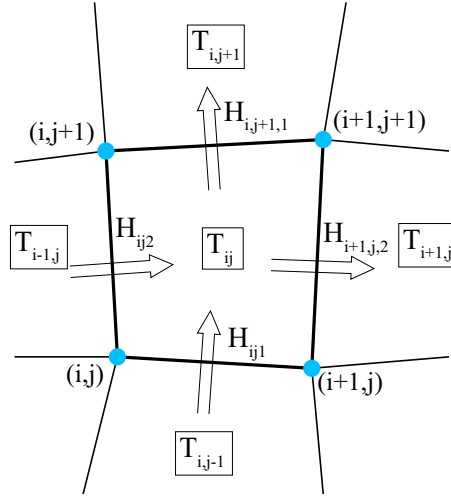
To be able to calculate the coefficients $D_{T,ij}$, we have to single out in Eq. (5.2) the explicit dependence on two face-adjacent temperatures:

$$H_{ij1} = -a_{ij1}T_{ij} + b_{ij1}T_{i,j-1} + \sum' c_{1\alpha}T_{\alpha}, \quad (5.3)$$

$$H_{ij2} = -a_{ij2}T_{ij} + b_{ij2}T_{i-1,j} + \sum'' c_{2\alpha}T_{\alpha}. \quad (5.4)$$

In the above expressions the summation \sum' is performed over all the participating temperatures other than T_{ij} and $T_{i,j-1}$, the summation \sum'' — over all participating temperatures other than T_{ij} and $T_{i-1,j}$. By their physical meaning, the coefficients a_{ijm} and b_{ijm} must be both non-negative. The algorithm for evaluation of the coefficients a_{ijm} and b_{ijm} depends on a particular scheme chosen for finite-difference approximation of the heat flux $\vec{h} = -\kappa \nabla T$.

Representation (5.3) leads us to the following expressions for the semi-implicit fluxes \tilde{H}_{α}

FIG. 5.1: Nomenclature of heat fluxes across the faces of cell (i, j) .

that should be used to calculate the semi-implicit deposition power $\tilde{W}_{T,ij}$:

$$\tilde{H}_{ij1} = -a_{ij1} \tau_{ij} + H_{ij1}, \quad (5.5)$$

$$\tilde{H}_{ij2} = -a_{ij2} \tau_{ij} + H_{ij2}, \quad (5.6)$$

$$\tilde{H}_{i,j+1,1} = b_{i,j+1,1} \tau_{ij} + H_{i,j+1,1}, \quad (5.7)$$

$$\tilde{H}_{i+1,j,2} = b_{i+1,j,2} \tau_{ij} + H_{i+1,j,2}. \quad (5.8)$$

From Eqs. (5.5)–(5.8) and (5.1) we find

$$D_{T,ij} = a_{ij1} + a_{ij2} + b_{i,j+1,1} + b_{i+1,j,2}. \quad (5.9)$$

Substituting Eqs. (5.1) and (5.9) into Eq. (4.6), we calculate τ_{ij} .

Correspondence with the code variables:

$a_{ijm} = \text{WWAT}(I,J,M)$ SSI coefficients;

$b_{ijm} = \text{WWBT}(I,J,M)$ SSI coefficients;

$H_{ijm} = \text{WWHT}(I,J,M)$ extensive, total heat fluxes;

$\tau_{ij} = \text{DTESSI}(I,J)$ intensive, temperature increments;

$T_{v,ij} = \text{TEMPV}(I,J)$ intensive, vertex temperatures;

5.2. Formulae for explicit deposition rates W_T and total fluxes H_{ijm}

Equation (5.1) expresses $W_{T,ij}$ in terms of face-centered total fluxes H_{ijm} , which, according to Eq. (A.20), are given by

$$H_{ijm} = \frac{\tilde{\kappa}_{f,ijm} R_{f,ijm}}{|J_{ijm}|} \left[\left(\vec{\lambda}_{v,ijm} \cdot \vec{\lambda}_{c,ijm} \right) \Delta T_{v,ijm} - |\vec{\lambda}_{v,ijm}|^2 \Delta T_{ijm} \right]. \quad (5.10)$$

Here

$$\Delta T_{ijm} = \begin{cases} T_{ij} - T_{i,j-1}, & m = 1, \\ T_{ij} - T_{i-1,j}, & m = 2, \end{cases} \quad (5.11)$$

$$\Delta T_{v,ijm} = \begin{cases} T_{v,i+1,j} - T_{v,ij}, & m = 1, \\ T_{v,i,j+1} - T_{v,ij}, & m = 2, \end{cases} \quad (5.12)$$

vector $\vec{\lambda}_{v,ij1}$ [or $\vec{\lambda}_{v,ij2}$] connects vertex (i, j) with vertex $(i + 1, j)$ [or $(i, j + 1)$], and vector $\vec{\lambda}_{c,ij1}$ [or $\vec{\lambda}_{c,ij2}$] connects cell center $(i, j - 1)$ [or $(i - 1, j)$] with cell center (i, j) ; see Fig. A.3. Jacobian J_{ijm} is given by the vector product $J_{ijm} = \vec{\lambda}_{v,ijm} \otimes \vec{\lambda}_{c,ijm}$, i.e. $|J_{ijm}| = 2 |A_{\Delta,ijm}^+ + A_{\Delta,ijm}^-|$. Face-centered radii $R_{f,ijm}$ are given by Eq. (A.16), and vertex-centered temperatures $T_{v,ij}$ are given by Eqs. (A.10) and (5.19). When evaluating the face-centered conduction coefficients $\tilde{\kappa}_{f,ijm}$, a flux limiting condition

$$\kappa_f |\nabla T| \leq h_l \quad (5.13)$$

is imposed as described in the following section; here $h_l > 0$ [erg cm⁻² s⁻¹] is the maximum allowed absolute value of the area-specific heat flux h . Physically, for electron heat conduction usually a formula is used

$$h_l = f_{inh} n_e T_e (T_e / m_e)^{1/2}, \quad (5.14)$$

where the dimensionless inhibition factor $f_{inh} \simeq 0.03$ –1.

5.3. Flux limiting

For each cell (i, j) a cell-centered flux limit $h_{l,ij}$ is defined. It is supposed to be calculated together with the cell-centered conduction coefficient $\kappa_{f,ij}$ in the thermodynamic part of the code. Then the flux-limit-corrected face-centered conduction coefficient $\tilde{\kappa}_{ijm}$ is calculated as

$$\tilde{\kappa}_{f,ijm} = \min \left\{ \kappa_{f,ijm}; \frac{h_{l,ijm}}{|g|_{ijm}} \right\}, \quad (5.15)$$

where $\kappa_{f,ijm}$ is given by Eqs. (A.21) and (A.22),

$$h_{l,ij1} = \begin{cases} h_{l,i,j-1}, & H_{ij1} > 0, \\ h_{l,ij}, & H_{ij1} \leq 0, \end{cases} \quad (5.16)$$

$$h_{l,ij2} = \begin{cases} h_{l,i-1,j}, & H_{ij2} > 0, \\ h_{l,ij}, & H_{ij2} \leq 0, \end{cases} \quad (5.17)$$

and

$$|g|_{ijm} = \frac{1}{|J_{ijm}|} \left[|\vec{\lambda}_{v,ijm}|^2 \Delta T_{ijm}^2 + |\vec{\lambda}_{c,ijm}|^2 \Delta T_{v,ijm}^2 - 2 (\vec{\lambda}_{v,ijm} \cdot \vec{\lambda}_{c,ijm}) \Delta T_{ijm} \Delta T_{v,ijm} \right]^{1/2} \quad (5.18)$$

is the absolute value of the face-centered temperature gradient $\vec{g} = (\nabla T)_f$ [see Eq. (A.18) below].

5.4. Formulae for coefficients D_T

Equation (5.9) expresses $D_{T,ij}$ in terms of face-centered coefficients a_{ijm} and b_{ijm} . To obtain explicit formulae for a_{ijm} and b_{ijm} from Eq. (A.20), we cast the interpolation scheme (A.10) for the vertex temperatures $T_{v,ij}$ in the form

$$T_{v,ij} = \mu_{1,ij} T_{ij} + \mu_{2,ij} T_{i-1,j} + \mu_{3,ij} T_{i-1,j-1} + \mu_{4,ij} T_{i,j-1}, \quad (5.19)$$

which complies with the CAVEAT convention about numbering of the four cells that surround a vertex (i, j) ; see Fig. 5.2. Coefficients $\mu_{\alpha,ij}$ are expressed in terms of the ξ, η values for the vertex (i, j) and the cell-centered conduction coefficients κ_{α} in the surrounding four cells, as given by Eq. (A.10), with a restriction $\mu_{\alpha,ij} > 0$ and a provision of $\mu_{1,ij} + \mu_{2,ij} + \mu_{3,ij} + \mu_{4,ij}$. As a result, we calculate

$$a_{ij1} = \frac{\tilde{\kappa}_{f,ij1} R_{f,ij1}}{|J_{ij1}|} \left[|\vec{\lambda}_{v,ij1}|^2 + \left(\vec{\lambda}_{v,ij1} \cdot \vec{\lambda}_{c,ij1} \right) (+\mu_{1,ij} - \mu_{2,i+1,j}) \right], \quad (5.20)$$

$$b_{ij1} = \frac{\tilde{\kappa}_{f,ij1} R_{f,ij1}}{|J_{ij1}|} \left[|\vec{\lambda}_{v,ij1}|^2 + \left(\vec{\lambda}_{v,ij1} \cdot \vec{\lambda}_{c,ij1} \right) (-\mu_{4,ij} + \mu_{3,i+1,j}) \right], \quad (5.21)$$

for face 1, and

$$a_{ij2} = \frac{\tilde{\kappa}_{f,ij2} R_{f,ij2}}{|J_{ij2}|} \left[|\vec{\lambda}_{v,ij2}|^2 + \left(\vec{\lambda}_{v,ij2} \cdot \vec{\lambda}_{c,ij2} \right) (+\mu_{1,ij} - \mu_{4,i,j+1}) \right], \quad (5.22)$$

$$b_{ij2} = \frac{\tilde{\kappa}_{f,ij2} R_{f,ij2}}{|J_{ij2}|} \left[|\vec{\lambda}_{v,ij2}|^2 + \left(\vec{\lambda}_{v,ij2} \cdot \vec{\lambda}_{c,ij2} \right) (-\mu_{2,ij} + \mu_{3,i,j+1}) \right], \quad (5.23)$$

for face 2.

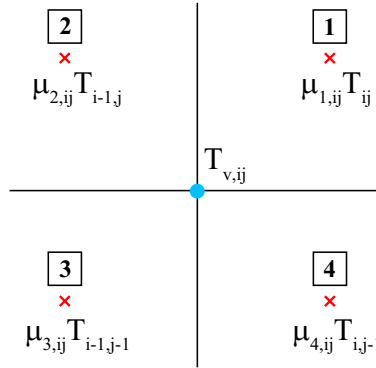


FIG. 5.2: Numbering of cell-centered quantities used to evaluate vertex temperatures T_{ij} .

5.5. Formulae for the energy corrections δ_T

Once the values of τ_{ij} have been determined for the entire mesh, we can calculate the energy correction $\bar{\delta}_{T,ij}$ [erg], which is lost at the current L-step in cell (i, j) , and which should be added at the next time step. The energy disbalance arises from the fact that the flux

\tilde{H}_{ij1} , given by Eq. (5.5), is not equal to the flux \tilde{H}_{ij1} , obtained from Eq. (5.7) by replacing j with $j - 1$. As a result, we have a situation where energy either disappears or is created at cell interfaces. Since the flux discrepancy is proportional to Δt , the energy correction will be proportional to Δt^2 .

If we denote the amount of energy disappearing at face $(i, j, 1)$ as $e_{d,ij1} \Delta t$ [erg], then from Eqs. (5.5) and (5.7) it is easy to calculate that

$$e_{d,ij1} = a_{ij1}\tau_{ij} + b_{ij1}\tau_{i,j-1}. \quad (5.24)$$

Similarly, if the amount of energy lost at face $(i, j, 2)$ is $e_{d,ij2} \Delta t$, then

$$e_{d,ij2} = a_{ij2}\tau_{ij} + b_{ij2}\tau_{i-1,j}. \quad (5.25)$$

Finally, the energy compensation for the next time step is found to be

$$\bar{\delta}_{T,ij} = [\chi_{ij1} e_{d,ij1} + \chi_{ij2} e_{d,ij2} + (1 - \chi_{i,j+1,1}) e_{d;i,j+1,1} + (1 - \chi_{i+1,j,2}) e_{d;i+1,j,2}] \Delta t, \quad (5.26)$$

where $0 \leq \chi_{ijm} \leq 1$ is the fraction of the energy $e_{d,ijm} \Delta t$ lost at face (i, j, m) , which is ascribed to the neighboring cell in the ‘‘forward’’ (i.e. along the normal $\vec{n}_{f,ijm}$) direction from this face (see Figs. 3.1 and 5.1). We assume that the weight χ_{ijm} is proportional to the total (not specific) heat capacity of the triangular region in cell (i, j) , made up of two vertices of face (i, j, m) and of the cell center $\vec{x}_{c,ij}$ (see Fig. 5.3); specific formulae are given in section 5.5.6. Similar weights are used in the original CAVEAT code to calculate the face-centered values $\kappa_{f,ijm}$ of the conduction coefficient κ .

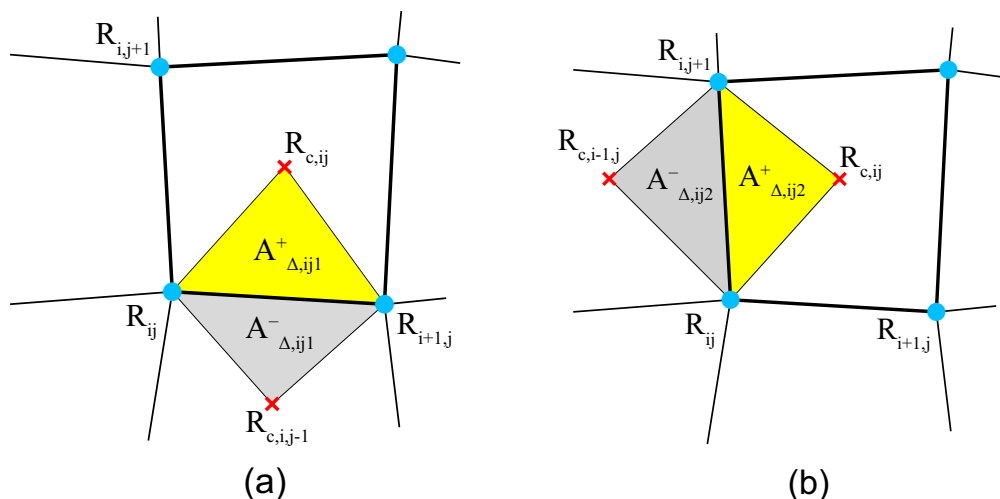


FIG. 5.3: Face-adjacent triangles used to calculate weights for splitting face-centered values between the two adjacent cells.

5.6. Weight coefficients for the energy correction

For the weight coefficients χ_{ijm} in Eq. (5.26) no formulae are available from the original CAVEAT version. We evaluate them as

$$\chi_{ijm} = \frac{C_{\Delta,ijm}^+}{C_{\Delta,ijm}^+ + C_{\Delta,ijm}^-} \quad (5.27)$$

with a restriction of

$$0 \leq \chi_{ijm} \leq 1; \quad (5.28)$$

here

$$C_{\Delta,ij1}^+ = \frac{1}{3} (R_{ij} + R_{i+1,j} + R_{c,ij}) A_{\Delta,ij1}^+ c_{V,ij} \rho_{ij}, \quad (5.29)$$

$$C_{\Delta,ij1}^- = \frac{1}{3} (R_{ij} + R_{i+1,j} + R_{c,i,j-1}) A_{\Delta,ij1}^- c_{V,i,j-1} \rho_{i,j-1}, \quad (5.30)$$

$$C_{\Delta,ij2}^+ = \frac{1}{3} (R_{ij} + R_{i,j+1} + R_{c,ij}) A_{\Delta,ij2}^+ c_{V,ij} \rho_{ij}, \quad (5.31)$$

$$C_{\Delta,ij2}^- = \frac{1}{3} (R_{ij} + R_{i,j+1} + R_{c,i-1,j}) A_{\Delta,ij2}^- c_{V,i-1,j} \rho_{i-1,j}, \quad (5.32)$$

R_{ij} is the radius R of vertex (i, j) , $R_{c,ij}$ is the radius of the (i, j) -cell center, $A_{\Delta,ijm}^+$ is the area of the (i, j, m) -face-adjacent triangle extending into cell (i, j) (see Fig. 5.3), and $A_{\Delta,ijm}^-$ is the area of the (i, j, m) -face-adjacent triangle extending into the neighboring cell [either $(i, j - 1)$ or $(i - 1, j)$]. Note that the quantities $C_{\Delta,ijm}^\pm$ may have negative values.

Correspondence with the code variables:

$R_{c,ij} =$	XC(I,J,IRADIAL)	cylindrical radius of the (i, j) -cell center;
$2\omega_m A_{\Delta,ijm}^+ =$	DETI(I,J,M)	area of the (i, j, m) -face adjacent quarter-triangle of cell (i, j) ; calculated in subroutine DETCALC;
$2\omega_m A_{\Delta,ijm}^- =$	DETJ(I,J,M)	area of the (i, j, m) -face adjacent quarter-triangle in the corresponding neighbor cell (i, j) ; calculated in subroutine DETCALC;
$\omega_m =$	HANDED	handedness of the mesh; calculated in subroutine MESHGEN;
$h_{l,ij} =$	HCONDL(I,J)	intensive, cell-centered thermal flux limits;
$\mu_{k,ij} =$	WWMU(I,J,K)	weight coefficients for evaluation of $T_{v,ij}$; $k = 1, 2, 3, 4$
$\chi_{ijm} =$	WWECWEI(I,J,M)	weight coefficients for evaluation of $\bar{\delta}_{T,ij} + \bar{\delta}_{r,ij}$

5.7. Time step control

The SSI algorithm requires a separate control of the time step Δt to ensure accuracy and stability of simulation. Because the energy correction $\delta_{ij} = \delta_{T,ij} + \delta_{r,ij}$ is taken from the previous step and cannot be reduced in the current hydro cycle, we need two separate constraints on the Δt value with two independent control parameters ε_0 and ε_1 [9]. We use the following two conditions for time step control at the SSI stage:

$$\left| \frac{(W_{T,ij} + W_{r,ij} + q_{ij} M_{ij}) \Delta t}{c_{V,ij} M_{ij} + \Delta t (D_{T,ij} + D_{r,ij})} \right| \leq (\varepsilon_0 - \varepsilon_1) (T_{ij} + T_s), \quad (5.33)$$

$$\left| \frac{\bar{\delta}_{T,ij} + \bar{\delta}_{r,ij}}{c_{V,ij} M_{ij}} \right| \leq \varepsilon_1 (T_{ij} + T_s), \quad (5.34)$$

where $T_s > 0$ is a problem-specific ‘‘sensitivity’’ threshold for temperature variations (not to be confused with the temperature floor value T_{flr}). Clearly, we must always have $\varepsilon_1 < \varepsilon_0$.

Condition (5.34) guarantees that in the next cycle the relative temperature variation due to the SSI energy correction δ_{ij} will not exceed ε_1 for any new value of $\Delta t > 0$. When applied together at each time step, the two conditions (5.33) and (5.34) guarantee that the total relative temperature variation $|\tau_{ij}|/(T_{ij} + T_s)$ [where τ_{ij} is given by Eq. (4.6)] never exceeds ε_0 . Note that by reducing the current time step Δt both conditions (5.33) and (5.34) can always be satisfied for any $\varepsilon_1 > 0$ and $\varepsilon_0 > \varepsilon_1$. Without condition (5.34) numerical instability was sometimes observed in certain test runs with flux limiting. Sufficiently accurate results have always been obtained with $\varepsilon_0 = 0.2$ and $\varepsilon_1 = 0.05$.

Correspondence with the code variables:

$\varepsilon_0 =$	EPS0SSI	principal time-step control parameter at the SSI stage; def = 0.2;
$\varepsilon_1 =$	EPS1SSI	secondary time-step control parameter at the SSI stage; def = 0.05;
$T_s =$	TEMPSNS	sensitivity threshold for T variation;
$T_{flr} =$	TEMPFLR	absolute minimum for T values; def = 10^{-30} ;

5.8. Boundary conditions

1. Physical boundaries

In the present code version only two types of boundary conditions for the heat conduction equation are foreseen: (i) zero flux (ITCONBC =1), and (ii) specified temperature (ITCONBC =2). By using Fig. 5.1 and the above formulae for fluxes, one easily verifies that these types of boundary conditions can be implemented by requiring that

if face $(i, j, 1)$ is part of physical boundary IB=1 (bottom), then

$$\begin{aligned} b_{ij1} = 0, \quad \chi_{ij1} = 1 & \quad \text{for ITCONBC} = 1,2, \\ a_{ij1} = 0, \quad H_{ij1} = 0 & \quad \text{for ITCONBC} = 1; \end{aligned} \quad (5.35)$$

if face $(i, j, 1)$ is part of physical boundary IB=2 (top), then

$$\begin{aligned} a_{ij1} = 0, \quad \chi_{ij1} = 0 & \quad \text{for ITCONBC} = 1,2, \\ b_{ij1} = 0, \quad H_{ij1} = 0 & \quad \text{for ITCONBC} = 1; \end{aligned} \quad (5.36)$$

if face $(i, j, 2)$ is part of physical boundary IB=3 (left), then

$$\begin{aligned} b_{ij2} = 0, \quad \chi_{ij2} = 1 & \quad \text{for ITCONBC} = 1,2, \\ a_{ij2} = 0, \quad H_{ij2} = 0 & \quad \text{for ITCONBC} = 1; \end{aligned} \quad (5.37)$$

if face $(i, j, 2)$ is part of boundary boundary IB=4 (right), then

$$\begin{aligned} a_{ij2} = 0, \quad \chi_{ij2} = 0 & \quad \text{for ITCONBC} = 1,2, \\ b_{ij2} = 0, \quad H_{ij2} = 0 & \quad \text{for ITCONBC} = 1; \end{aligned} \quad (5.38)$$

and by setting corresponding values of the boundary temperature TEMPBC(kprt,ib,iblk) and conduction coefficient TCCOFBC(kprt,ib,iblk). The above boundary values for weight coefficients χ_{ijm} ensure that all the SSI energy correction $e_{d,ijm}\Delta t$, generated at a boundary interface, will be deposited in the physical mesh domain.

2. Block interfaces

At block interfaces all the above quantities a_{ijm} , b_{ijm} , H_{ijm} , and χ_{ijm} are calculated by using ghost-cell data transferred from the corresponding neighbor-block. After the quantities τ_{ij} are calculated, they should be passed through the inter-block communication procedure as well. Special treatment is needed however for three-block meeting points.

Three-block meeting point. If the “primary” corner of a block boundary IB (i.e. the left-hand corner when the block body is above this boundary) is a 3-block meeting point with no void left (a 3-bk point), the flag $\text{I3BK}(\text{IB},\text{IBLK})$ is set equal to 1. In this case the corresponding corner ghost cell in each of the three meeting blocks is degenerate and has zero volume, whereas next-to-corner ghost cells get their cell-centered values from the corresponding physical cells of the meeting blocks. Although “reasonable” (i.e. interpolated from the two neighboring cells) cell-center values of physical quantities are nevertheless assigned to the degenerate corner cell, the vertex temperature T_v at the meeting point, calculated as described in section A 1, turns out to be different in each of the three meeting blocks. As a result, the conduction scheme becomes non-conservative with respect to energy.

Example: if $\text{I3BK}(\text{IB},\text{IBLK})=1$, then logically different ghost vertices $(0,0)$, $(0,1)$, and $(1,0)$ coincide in physical space and lie on the common boundary between the two other meeting blocks; see Fig. 5.4.

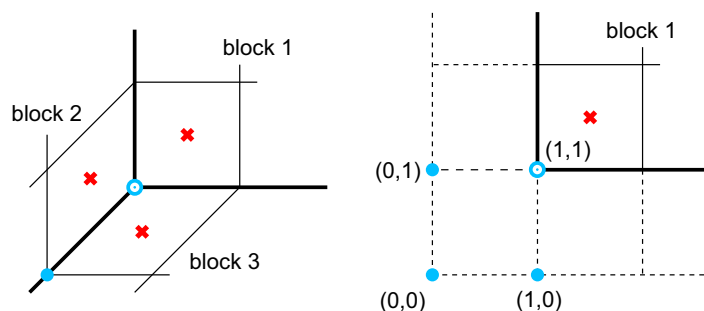


FIG. 5.4: Ghost cells around a 3-block meeting point.

To restore the conservativeness of the algorithm, a three-point interpolation scheme should be used for the T_v value at the 3-bk point. To this end, the general interpolation formula

$$T_{v,ij} = \mu_{1,ij} T_{ij} + \mu_{2,ij} T_{i-1,j} + \mu_{3,ij} T_{i-1,j-1} + \mu_{4,ij} T_{i,j-1} = \sum_{\alpha=1}^4 \mu_{\alpha} T_{\alpha} \quad (5.39)$$

is cast in the form

$$T_{v,ij} = \sum_{k=0}^3 \mu_{\alpha_k} T_{\alpha_k}, \quad (5.40)$$

where $\alpha_k = \{\alpha_0, \alpha_1, \alpha_2, \alpha_3\}$ is such a permutation of indices $\alpha = \{1, 2, 3, 4\}$ that α_0 corresponds to the corner ghost cell in each of the three meeting blocks. Then the value of μ_{α_0}

is set equal to zero, while the values of μ_{α_k} , $k = 1, 2, 3$ are calculated from a linear 3-point interpolation between the remaining three cell centers $\alpha_1, \alpha_2, \alpha_3$:

$$\mu_{\alpha_0} = 0, \quad (5.41)$$

$$\mu_{\alpha_1} = 1 - \mu_{\alpha_2} - \mu_{\alpha_3}, \quad (5.42)$$

$$\mu_{\alpha_2} = \frac{(y_{\alpha_3} - y_{\alpha_1})(x_v - x_{\alpha_1}) - (x_{\alpha_3} - x_{\alpha_1})(y_v - y_{\alpha_1})}{det}, \quad (5.43)$$

$$\mu_{\alpha_3} = \frac{-(y_{\alpha_2} - y_{\alpha_1})(x_v - x_{\alpha_1}) + (x_{\alpha_2} - x_{\alpha_1})(y_v - y_{\alpha_1})}{det}, \quad (5.44)$$

$$det = (x_{\alpha_2} - x_{\alpha_1})(y_{\alpha_3} - y_{\alpha_1}) - (y_{\alpha_2} - y_{\alpha_1})(x_{\alpha_3} - x_{\alpha_1}). \quad (5.45)$$

Since the 3-point linear interpolation is uniquely defined (when the three cell centers surrounding the 3-bk vertex make up a non-degenerate triangle) and depends on parameters in the physical cells only, it yields the same 3-bk vertex temperature in each of the three meeting blocks. In the numerical algorithm the coefficients μ_{α_k} are additionally restricted to lie within the interval $0 \leq \mu_{\alpha_k} \leq 1$.

Three-block-void meeting point. In this case three blocks meet at a joint corner vertex, but an adjoining void sector is also present (a 3-vd point). Here again, to preserve the conservativeness of the algorithm, a 3-point interpolation scheme should be used for the vertex temperature at the common corner of the 3 meeting blocks; see Fig. 5.5. Of the 3 meeting blocks, the flag I3VD(IB,IBLK) is set equal to 1 only for the two diagonally opposite blocks which border on void. In each of these two blocks the next-to-the-corner ghost cell, which borders on void, is excluded from the 4-point interpolation scheme for the vertex temperature at the physical corner (unshaded in Fig. 5.5).

Interblock communication: 3-block-void meeting point

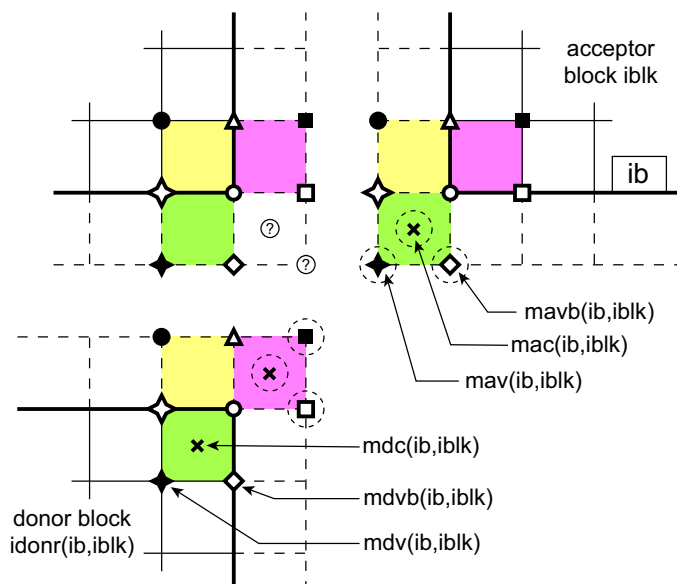


FIG. 5.5: Ghost cells around a 3-block-void meeting point.

For the third central block, which does not border on void, a new flag I3DD(IB,IBLK) is set equal to 1. In this block the corner ghost cell (unshaded in Fig. 5.5) is excluded from

the interpolation scheme. Note that the values of various quantities in this ghost cell are spurious.

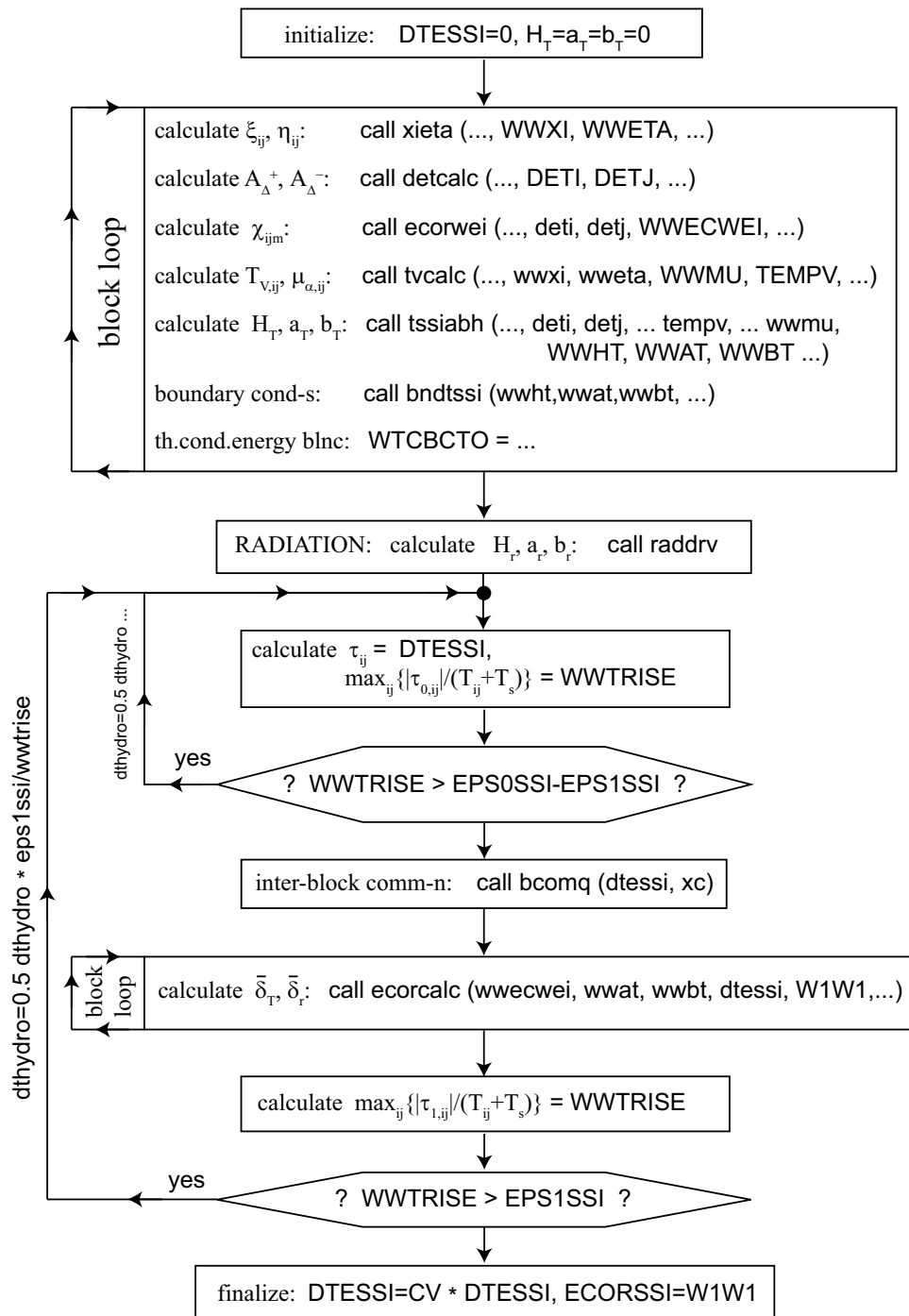


FIG. 5.6: Block-scheme of the SSI algorithm in the CAVEAT-TR code.

6. TEST PROBLEMS

6.1. Problem 1: one-dimensional non-linear heat wave

In the first series of tests, propagation of a planar non-linear heat wave was simulated using both the explicit and the SSI algorithms. We assume a power-law form of the heat conduction coefficient,

$$\kappa(\rho, T) = \kappa_0 T^n, \quad (6.1)$$

in the one-dimensional planar heat conduction equation

$$\rho c_V \frac{\partial T}{\partial t} = \frac{\partial}{\partial x} \left(\kappa \frac{\partial T}{\partial x} \right). \quad (6.2)$$

Here c_V is the mass-specific heat capacity at constant volume; the quantities κ_0 , c_V , and ρ are assumed to be constant.

To test the code, the heat wave propagation in a half-space with a constant driving temperature $T = T_0$ at the boundary plane $x = 0$ is considered. By transforming to the self-similar variable

$$\xi = \left(\frac{n+1}{2} \frac{\rho c_V}{\kappa_0 T_0^n} \right)^{1/2} \frac{x}{\sqrt{t}}, \quad (6.3)$$

Eq. (6.2) is reduced to

$$\frac{d^2 \tau^{n+1}}{d\xi^2} + \xi \frac{d\tau}{d\xi} = 0, \quad \text{where } \tau = \frac{T}{T_0}. \quad (6.4)$$

The boundary conditions are $\tau = 1$ at $\xi = 0$, and $\tau = \tau^n (d\tau/d\xi) = 0$ at an unknown front position $\xi = \xi_0$, which is to be determined in the process of solution of the boundary value problem. The condition at $\xi = \xi_0$ expresses the fact that the heat flux, proportional to $\tau^n (\partial\tau/\partial\xi)$, is a continuous function of ξ .

Equation (6.4) is readily solved numerically by transforming to a new independent function $y = \tau^n$. The position of the thermal front is given by

$$x_f = \xi_0 \left[\frac{2\kappa_0 T_0^n}{(n+1)\rho c_V} t \right]^{1/2}. \quad (6.5)$$

In tests 1a, 1b and 1c below the heat wave was simulated for $n = 3$. In this case numerical integration of Eq. (6.4) yields an eigenvalue $\xi_0 = 1.231172$.

1a. Planar heat wave on a 1D mesh launched by a thermal reservoir

Here we test heat wave propagation from a hot region inside the computational volume, which is a rectangular block in cartesian coordinates with reflective boundary conditions. In the x -direction the computational region is divided into two parts. The first part serves as a thermal reservoir to impose the constant temperature on the left boundary of the second part. To this end, the heat capacity and the conduction coefficient of the reservoir gas are chosen to be much larger than the corresponding values in the propagation region. Also, the gas parameters are adjusted such that the hydrodynamical time scale is 10^8 times longer than the thermal time scale, i.e. the gas remains practically at rest as the heat wave propagates. Sample runs have shown that the number of cells along the y -axis affects the results by no more than $\simeq 2 \times 10^{-4}$.

The numerical problem has been configured as follows:

$$\begin{array}{ccc}
 x = -1 & & 0 & & 1 \\
 \rho = 1, T = 1, \gamma = 1.001 & & & & \rho = 1, T = 0, \gamma = 2 \\
 \kappa = 10^{11}(\text{explicit}), 10^{12}(\text{SSI}) & & & & \kappa = 10^8 T^3 \\
 \text{NCELL}(1,1,1)=25 & & & & \text{NCELL}(2,1,1)=100
 \end{array}$$

In both regions the ideal-gas equation of state is assumed,

$$p = (\gamma - 1)\rho\epsilon = \rho T, \quad T = \frac{\epsilon}{c_V}, \quad c_V = \frac{1}{\gamma - 1}. \quad (6.6)$$

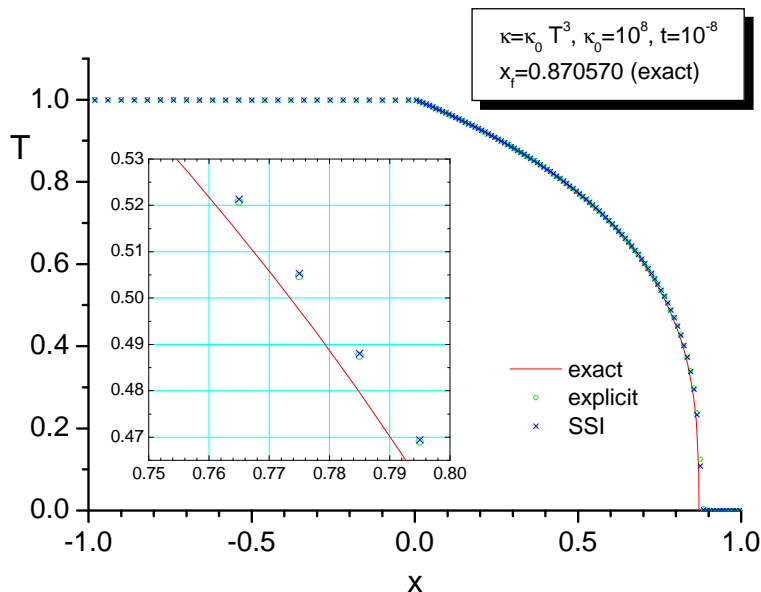


FIG. 6.1: Results for test problem 1a, where the heat wave propagates from an adjoining thermal reservoir at $x < 0$.

Figure 6.1 shows the temperature profiles at $t = 10^{-8}$ calculated using the explicit and the SSI schemes, as well as the exact solution. The insert shows a blow-up of the region around $T = 0.5T_0$ ($T_0 = 1$). The CAVEAT solutions follow the exact profile with a good accuracy. The result for the explicit scheme was obtained in about $N_{cyc} \simeq 500\,000$ time steps. In this case, the time step is limited not in the heat wave region but in the reservoir, where the conduction coefficient is large, $\kappa = 10^{11}$. The SSI calculations were done for $\kappa = 10^{12}$ inside the thermal reservoir and required $N_{cyc} = 3\,171$ time steps for $\varepsilon_0 = 0.2$, $\varepsilon_1 = 0.02$ in the criteria (5.33) and (5.34). The values $\text{TEMPFLR} = 1.D-30$, $\text{TEMPSNS} = 1.D-3$ have been assumed for the temperature “floor” T_{flr} and sensitivity threshold T_s .

TABLE I: Reference values from the exact, explicit and SSI solutions in test 1a.

reference quantity	exact	explicit	SSI
T at $x = 0.775$	0.4974	0.5047	0.5053
wave front x_f	0.870570	0.8755 ± 0.0005	0.8750 ± 0.0005

A quantitative comparison between the exact, explicit and SSI solutions is given in Table I for temperatures values at $T \approx 0.5T_0$, and for the wave front position x_f . One sees that both numerical solutions reproduce the exact temperature profile to an accuracy of about 1.5%; the front position x_f is calculated with a relative error of $\simeq 0.7\%$. These results clearly demonstrate that the SSI method allows to solve heat conduction problems 10–100 times faster than the explicit scheme, preserving essentially the same accuracy. It becomes particularly useful when certain regions of the computational domain have very high values of the conduction coefficient and small sizes of mesh cells.

1b. Planar heat wave on a 1D mesh launched by a fixed boundary temperature

Here we test propagation of a heat wave from an external heat source represented by an appropriate (specified temperature) boundary condition. The computational region consists of a single part. At the left boundary the temperature is fixed at $T = T_0 = 1$, ITCONBC(1,3) = 2, TEMPBC(1,3,1)=1.D0, TCCOFBC(1,1,3)=1.D8; the boundary value of the conduction coefficient is set equal to TCCOFBC(1,1,3)=1.D8. Along all the external boundaries of the computational region a zero-pressure boundary condition is prescribed for hydrodynamics, PBC(1,3,1)=0.D0. The width of the boundary “ghost” cells has been chosen much smaller than the width of the adjacent physical cells, GHWIDTH=1.D-12.

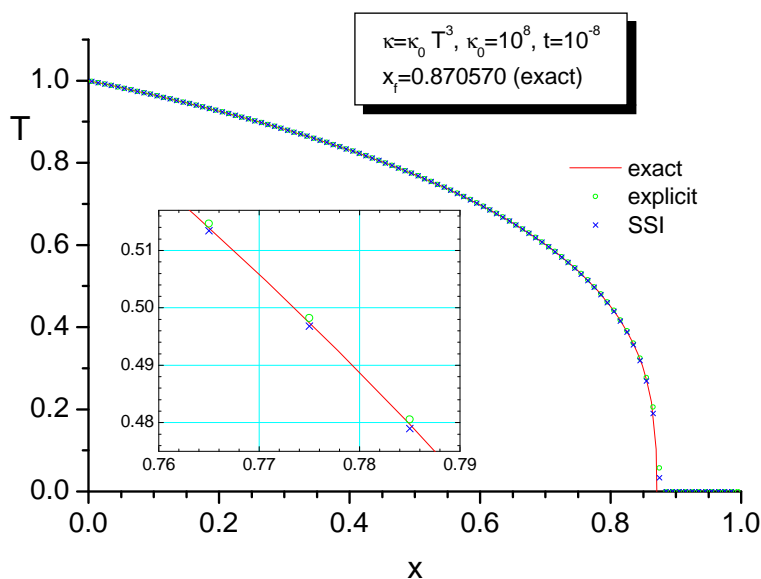


FIG. 6.2: Results for test problem 1b, where the heat wave is launched by a fixed boundary temperature $T = T_0 = 1$ at $x = 0$.

Figure 6.2 shows a comparison of the CAVEAT temperature profiles with the exact solution at $t = 10^{-8}$. For a more accurate quantitative comparison one can use reference values from table II. The SSI run, shown in Fig. 6.2, required $N_{cyc} = 3052$ time steps with the values of $\varepsilon_0 = 0.2$, $\varepsilon_1 = 0.02$, $T_{flr} = 10^{-30}$, $T_s = 10^{-3}$.

The CAVEAT results presented in Fig. 6.2 and table II demonstrate a significantly better accuracy than in the previous case 1a: deviations from the exact solution for both the explicit and the SSI runs do not exceed 0.1–0.2%. Such accuracy is quite impressive for a non-linear

TABLE II: Reference values from the exact, explicit and SSI solutions in test 1b.

reference quantity	exact	explicit	SSI
T at $x = 0.775$	0.4974	0.4982	0.4968
wave front x_f	0.870570	0.8720 ± 0.0005	0.8700 ± 0.0005

wave extending over less than 100 discrete mesh cells. In this way we verify (i) that the spatial part of the SSI algorithm is of sufficiently high accuracy, and (ii) that the boundary condition for heat conduction is correctly implemented in the code.

1c. Planar heat wave on a skewed 2D mesh launched by a fixed boundary temperature

This example tests propagation of a 1D non-linear planar heat wave on a 2D skewed mesh. The mesh used in this test consists of four quadrangular blocks and is shown in Fig. 6.3. The heat wave propagates from the bottom boundary of block 3, where the temperature is fixed at $T = T_0 = 1$. The mesh has 50 cells along the y -axis (direction of wave propagation), and 25 cells in the perpendicular x direction. Here we test the numerical scheme in the situation where the symmetry of the solution (planar) differs significantly from the symmetry of the numerical grid. The heat conduction coefficient and the equation of state are the same as in test 1b.

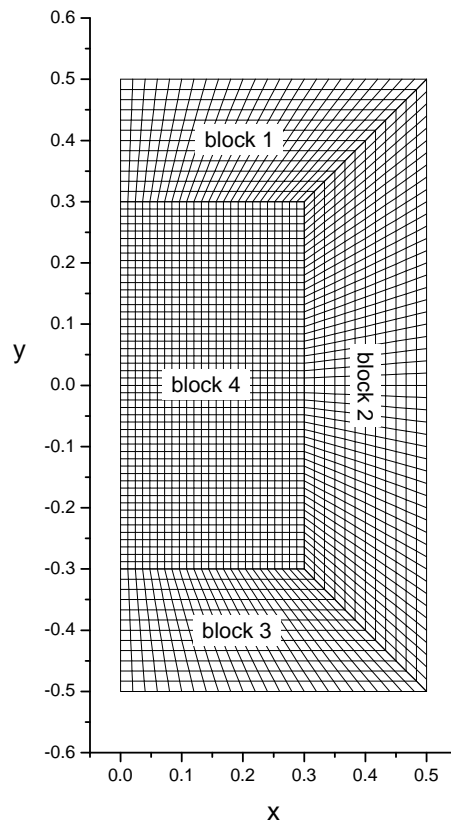


FIG. 6.3: Skewed 4-block mesh for test problem 1c.

The numerical results are compared with the exact solution in Fig. 6.4 for $t = 10^{-8}$, where temperature profiles along the $x = 0.006$ and $x = 0.475$ lines are plotted. The insert is a blow-up of the region around $T = 0.5T_0$. The SSI run, shown in Fig. 6.4, required $N_{cyc} = 3554$ time steps with the values of $\varepsilon_0 = 0.2$, $\varepsilon_1 = 0.02$, $T_{flr} = 10^{-30}$, $T_s = 10^{-3}$. Table III gives temperatures at $T \approx 0.5T_0$ and the positions of the front for the both profiles.

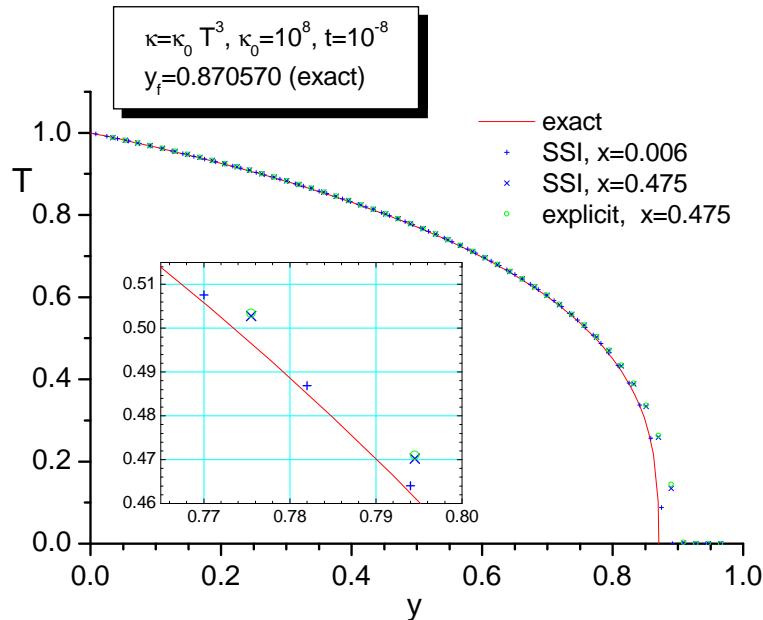


FIG. 6.4: Results for test problem 1c: a planar heat wave is launched by a fixed boundary temperature $T = T_0 = 1$ at $y = 0$. Line-out profiles for $x = 0.006$ and $x = 0.475$ are plotted.

At $x = 0.006$ the cells are practically rectangular, and, similar to test 1b, an excellent agreement is observed with the exact solution. For this case only the SSI results are plotted in Fig. 6.4. At $x = 0.475$ a very close agreement is observed between the SSI and explicit results, but both differ from the exact solution by about 1.2–1.8%. The observed 1–2% deviation from the exact solution must be caused by the propagation of a planar heat front through skewed quadrangles of the numerical grid.

TABLE III: Reference values from the exact, explicit and SSI solutions in test 1c.

reference quantity	exact	explicit	SSI
$x = 0.006$			
T at $y = 0.77$	0.5059	–	0.5076
wave front	0.870570	–	0.872 ± 0.0005
$x = 0.475$			
T at $y = 0.7755$	0.4966	0.5034	0.5028
wave front	0.870570	0.887 ± 0.0005	0.8865 ± 0.0005

With this test, a correct implementation of the inter-block communication scheme has also been validated, both along the block edges and at different types of the corner meeting points.

6.2. Problem 2: a spherical heat wave from an instantaneous point source

Here we consider a spherically symmetric non-linear heat wave governed by the equation

$$\rho c_V \frac{\partial T}{\partial t} = \frac{1}{r^2} \frac{\partial}{\partial r} \left(r^2 \kappa \frac{\partial T}{\partial r} \right), \quad (6.7)$$

where the conduction coefficient κ is given by Eq. (6.1). The mass-specific heat capacity c_V and the gas density ρ are supposed to be constant. We assume that at time $t = 0$ a finite amount of energy E is instantaneously released in the center at $r = 0$. The solution to this problem is fully analytical and can, for example, be found in Ref. [10, Ch. X]. It has the form

$$T(r, t) = T_c \left(1 - \frac{r^2}{r_f^2} \right)^{1/n}, \quad (6.8)$$

where the wave front radius is given by

$$r_f = r_f(t) = \xi_1 \left(\frac{\kappa_0 t}{\rho c_V} Q^n \right)^{\frac{1}{3n+2}}, \quad (6.9)$$

and the central temperature is

$$T_c = T_c(t) = \left[\frac{n \xi_1^2}{2(3n+2)} \right]^{1/n} Q^{\frac{2}{3n+2}} \left(\frac{\rho c_V}{\kappa_0 t} \right)^{\frac{3}{3n+2}}. \quad (6.10)$$

Here the parameter Q is defined as

$$Q = \frac{E}{\rho c_V} = 4\pi \int_0^\infty T r^2 dr, \quad (6.11)$$

and the dimensionless constant

$$\xi_1 = \left[\frac{3n+2}{2^{n-1} n \pi^n} \right]^{\frac{1}{3n+2}} \left[\frac{\Gamma\left(\frac{5}{2} + \frac{1}{n}\right)}{\Gamma\left(1 + \frac{1}{n}\right) \Gamma\left(\frac{3}{2}\right)} \right]^{\frac{n}{3n+2}} \quad (6.12)$$

is obtained by solving the corresponding eigenvalue problem.

For numerical test runs we select a particular case of $n = 2$ because, on the one hand, this is already quite close to the physically most interesting case of the Spitzer conductivity with $n = 5/2$, while, on the other hand, the parameter ξ_1 can be very simply calculated as

$$\xi_1 = \frac{2^{7/8}}{\sqrt{\pi}} = 1.03472826. \quad (6.13)$$

If, further, we assume $Q = 1$ and introduce a notation

$$\bar{t} = \frac{\kappa_0 t}{\rho c_V}, \quad (6.14)$$

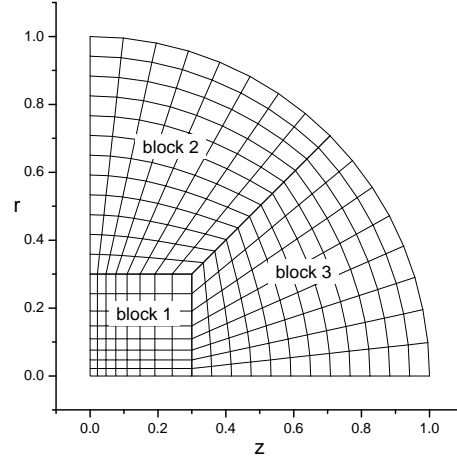
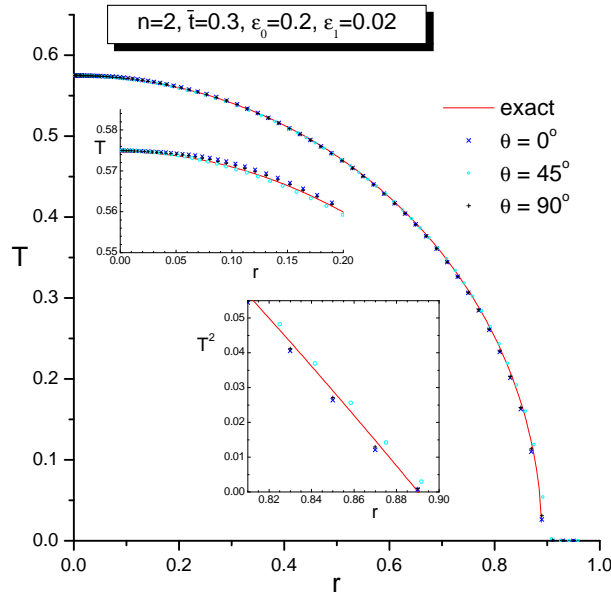


FIG. 6.5: Skewed 3-block mesh for test problem 2.

FIG. 6.6: Temperature profiles for test problem 2: a spherical heat wave is launched at $t = 0$ by an instantaneous point source in the center. Three line-outs, corresponding to the polar angle values $\theta = 0, 45^\circ$, and 90° , are compared with the exact solution at $\bar{t} = 0.3$.

we obtain the following exact values of r_f and T_c :

$$\begin{aligned} r_f &= \xi_1 \bar{t}^{1/8} = 0.890\,1567, \quad \bar{t} = 0.3, \\ T_c &= \frac{\xi_1}{2\sqrt{2}} \bar{t}^{-3/8} = 0.574\,5937, \quad \bar{t} = 0.3. \end{aligned} \quad (6.15)$$

Clearly, by choosing a sufficiently large value of κ_0 , we can always make the hydrodynamic time scale to be arbitrarily long compared to the thermal time scale $t \simeq \rho c_V / \kappa_0$.

Numerical simulation of the above stated problem was conducted in the cylindrical r, z coordinates (one of the two principal options for the intrinsic coordinate system in the

TABLE IV: Numerical results for test problem 2 that are to be compared with the following exact values: $T_c = 0.5745937$, $r_f = 0.8901567$.

ε_0	ε_1	T_c	$r_{f,z}$	$r_{f,45}$	$r_{f,r}$	N_{cyc}
0.1	0.01	0.5736	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	21 880
	0.02	0.5735	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	20 080
	0.04	0.5753	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	23556
	0.05	0.5750	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	27 546
	0.08	0.5735	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	68 701
0.2	0.01	0.5731	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	17 310
	0.02	0.5750	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	13 502
	0.03	0.5704	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	12 252
	0.04	0.5743	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	11 819
	0.05	0.5698	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	11 714
	0.08	0.5790	0.887 ± 0.0005	0.896 ± 0.0005	0.8885 ± 0.0005	12 806

CAVEAT code). Since the symmetry of the tested solution is quite different from the symmetry of the adopted coordinate system, the present test provides a good check for possible spurious numerical effects that might arise along the z -axis, where the coordinate singularity occurs. To aggravate the situation, we used a 3-block skewed mesh (see Fig. 6.5) which has no circular symmetry in the r, z plane as well. Because of the rotational symmetry around the z -axis, our mesh represents one half of a sphere of radius 1.

A progressively increasing cell size was used in the central block 1, so that $\Delta z_{11} = \Delta r_{11} = 2 \times 10^{-3}$ for the first cell in the center, and $\Delta z = \Delta r = 2 \times 10^{-2}$ along, respectively, the $\theta = 0$ (z -axis, $r = 0$) direction in block 3, and the $\theta = 90^\circ$ (radial, $z = 0$) direction in block 2. The condition $Q = 1$ was ensured by setting the initial value of the mass-specific internal energy in the central cell equal to

$$e_{11} = \frac{c_V}{4\pi V_{11}}, \quad (6.16)$$

where $V_{11} = \frac{1}{2}\Delta r_{11}^2 \Delta z$ is the volume (per steradian of the azimuth angle) of cell $(i, j) = (1, 1)$. Because the volume of the central cell V_{11} comprises a negligible fraction of 1.2×10^{-8} of the total volume of the simulated hemisphere ($1/3$ per steradian of the azimuth angle), no noticeable effects can be expected from the non-point-like form of the initial energy deposition.

The results of numerical simulations with the SSI algorithm are presented in Fig. 6.6 and table IV. One of the goals of these simulations was to determine an optimal combination of the two SSI accuracy parameters ε_0 and ε_1 , which would ensure a sufficiently good accuracy with not too many time steps N_{cyc} . From table IV one infers that the values $\varepsilon_0 = 0.2$, $\varepsilon_1 = 0.02$ are a good combination: the temperature profiles along all three line-outs in Fig. 6.6 are described to an accuracy of about 0.2% — which is quite impressive for a skewed mesh with a cell size of $\Delta r = \Delta z = 0.02$. The optimal ratio between the two accuracy parameters appears to be $\varepsilon_1/\varepsilon_0 = 0.1$ – 0.2 . For larger relative values of ε_1 slightly

non-monotonic temperature profiles are observed near the center $r = z = 0$. No spurious numerical effects along the rotational z -axis have been detected. A relatively large number $N_{cyc} \gtrsim 10\,000$ of time steps in all numerical runs is due to the fact that the temperature contrast between the initial central value, $T_{11} \approx 2 \times 10^7$, and the “sensitivity” threshold $T_s = 10^{-3}$ is more than 10 orders of magnitude.

6.3. Problem 3: a planar shock wave with heat conduction

Here we consider a planar shock wave in a medium with non-linear heat conduction. Let a steady-state planar shock propagate with a velocity D along x direction. An ideal-gas equation of state (6.6) is assumed. The gas in front of the shock is at rest and has the initial temperature $T = T_0 = 0$ and density $\rho = \rho_0$. In the reference frame comoving with the shock front the equations of mass, momentum and energy balance are [10, Ch. VII]

$$\rho u = -\rho_0 D, \quad (6.17)$$

$$p + \rho u^2 = \rho_0 D^2, \quad (6.18)$$

$$\rho u \left(\frac{\gamma}{\gamma - 1} T + \frac{1}{2} u^2 \right) - \kappa \frac{dT}{dx} = -\frac{1}{2} \rho_0 D^3. \quad (6.19)$$

After we introduce a new dimensionless variable

$$\eta = \frac{\rho_0}{\rho} \quad (6.20)$$

Eqs. (6.17) and (6.18) yield

$$u = -D\eta, \quad (6.21)$$

$$T = D^2 \eta(1 - \eta), \quad (6.22)$$

whereas the energy equation (6.19) becomes

$$\kappa \frac{dT}{dx} = \frac{1}{2} \rho_0 D^3 (1 - \eta) \left(1 - \frac{\gamma + 1}{\gamma - 1} \eta \right). \quad (6.23)$$

Now, the shock structure can be fully resolved in parametric form, with η treated as an independent parameter. Further on, we consider a specific case of $\gamma = 5/3$ and

$$\kappa = \kappa_0 T^2. \quad (6.24)$$

From Eqs. (6.21)-(6.23) one infers that the unperturbed gas first passes through a thermal precursor, where η changes continuously from $\eta = \eta_0 = 1$ to $\eta = \eta'_1 = 2/(\gamma + 1) = 3/4$, and the gas is compressed to $\rho = \rho'_1 = \rho_0/\eta'_1 = (4/3)\rho_0$; the temperature rises from $T = T_0 = 0$ to $T = T'_1 = T_1 = D^2 \eta'_1(1 - \eta'_1) = (3/16)D^2$; see Fig. 6.7. Then the gas passes through an isothermal ($T_1 = T'_1$) density jump by a factor $\rho_1/\rho'_1 = \eta'_1/\eta_1 = 2/(\gamma - 1) = 3$; the η parameter jumps from $\eta'_1 = 2/(\gamma + 1) = 3/4$ to $\eta = \eta_1 = (\gamma - 1)/(\gamma + 1) = 1/4$.

By substituting Eqs. (6.22) and (6.24) into Eq. (6.23) and integrating, one obtains

$$x = \frac{\kappa_0}{\rho_0} \left(\frac{D}{4} \right)^3 \left[-16\eta^4 + \frac{80}{3}\eta^3 - 6\eta^2 - 3\eta - \frac{9}{16} - \frac{3}{4} \ln \left(2\eta - \frac{1}{2} \right) \right], \quad (6.25)$$

where we have assumed $x = 0$ at the density jump. Equations (6.20)-(6.22) and (6.25) give a full solution in a parametric form for the shock wave structure; in Eqs. (6.22) and (6.25) the parameter η varies within the interval $3/4 \leq \eta \leq 1$, where $x = 0$ and $T = T_1$, and $\eta = 1$, where $x = x_f$ and $T = T_0 = 0$. For numerical simulations we choose the values $\kappa_0 = \rho_0 = 1$ and $D = 4$; then $\rho_1 = 4$, $T_1 = 3$, $p_1 = 12$, and

$$x_f = \frac{53}{48} - \frac{3}{4} \ln \frac{3}{2} = 0.8000678. \quad (6.26)$$

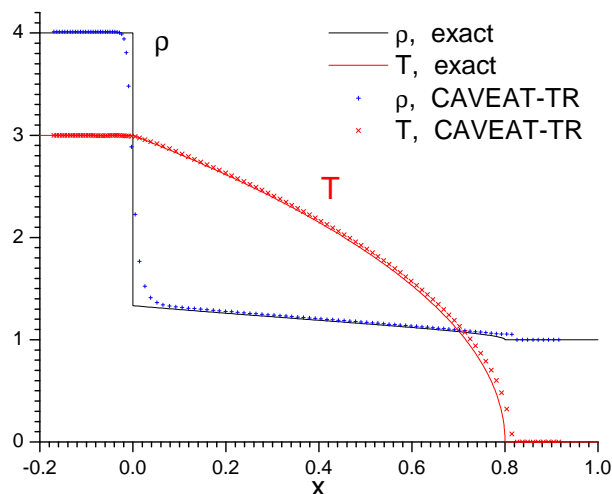


FIG. 6.7: Temperature and density profiles for test problem 3: a planar shock front with heat conduction.

In numerical simulation the shock front was launched by a fixed boundary pressure $p_1 = 12$ (rigid piston). Since initially the density jump has no thermal precursor, the shock front has to propagate over a certain distance before the steady-state profiles are established. To make this relaxation distance as short as possible, we assign also a fixed boundary temperature $T = T_1 = 3$ and a conduction coefficient $\kappa = \kappa_0 = 1$ at the piston boundary: the resulting heat inflow through the piston surface provides quick replenishment of the energy that escapes into the precursor region. The computational domain is a simple single-block rectangular region at $-10 \leq x \leq +1$ divided into 300 cells with a progressively diminishing length down to $\Delta x = 0.01$ at $x = +1$. The shock wave is launched at $x = -10$ at time $t = 0$. By the time $t = 2.5$ the shock front arrives at $x = 0$, and Fig. 6.7 shows the density and temperature profiles at this moment. More precisely, the profiles are shifted by $\Delta x = +0.035$ to place the density jump at $x = 0$ exactly (this small discrepancy in the distance traveled must be due to the non-equilibrium initial shock front structure). This CAVEAT run required 17 333 time steps with $\varepsilon_0 = 0.2$, $\varepsilon_1 = 0.02$, and $T_s = 10^{-3}$. In Fig. 6.7 one sees that the exact solution is reasonably well reproduced, with a typical error for the temperature profile at the front end of the precursor being $\simeq 2\%$.

APPENDIX A: CAVEAT INTERPOLATION SCHEMES

1. Vertex temperatures

The original CAVEAT version contains an explicit algorithm for heat conduction, which, in addition to cell-centered temperatures T , uses vertex-centered values T_v to evaluate the face-centered temperature gradients. The values of T_v are obtained by interpolation from the surrounding four cell-centered values of T . We take this scheme over to CAVEAT-TR without changes.

Besides the mesh geometry, the CAVEAT algorithm for evaluating T_v uses also cell-centered values of the conduction coefficient κ to calculate the corresponding interpolation coefficients. *Being quite appropriate for heat conduction proper, this is questionable for radiation transport.* To illustrate the idea, first consider the simpler 1D case.

a. 1D interpolation

Assume that the temperature values T and T_+ are associated with the cell centers x_c and x_{c+} , respectively; see Fig. A.1. Then, if we want to evaluate the temperature T_v at a mesh node x_v ($x_c \leq x_v \leq x_{c+}$), the simplest algorithm would be based on the assumption of a continuous gradient ∇T at point x_v :

$$\frac{T_v - T}{x_v - x_c} = \frac{T_+ - T_v}{x_{c+} - x_v}, \quad (\text{A.1})$$

which leads us to the expression

$$T_v = \frac{(x_{c+} - x_v)T + (x_v - x_c)T_+}{x_{c+} - x_v + x_v - x_c}. \quad (\text{A.2})$$

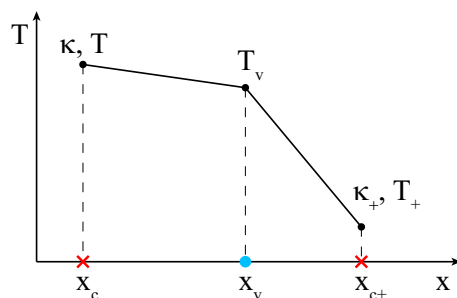


FIG. A.1: 1D illustration of the temperature interpolation algorithm.

However, in many cases, when the conduction coefficient κ is either a strong function of temperature or simply discontinuous, it will be more physical to assume that the flux $\kappa\nabla T$ is continuous rather than ∇T . In our 1D case this leads to

$$\kappa \frac{T_v - T}{x_v - x_c} = \kappa_+ \frac{T_+ - T_v}{x_{c+} - x_v} \quad (\text{A.3})$$

instead of (A.2); from Eq. (A.3) we calculate

$$T_v = \frac{\kappa(x_{c+} - x_v) T + \kappa_+(x_v - x_c) T_+}{\kappa(x_{c+} - x_v) + \kappa_+(x_v - x_c)}. \quad (\text{A.4})$$

An obvious generalization of Eq. (A.4) to a multidimensional case from a “normal” (continuous gradient) interpolation scheme

$$T_v = \frac{\sum_{\alpha} \mu_{\alpha} T_{\alpha}}{\sum_{\alpha} \mu_{\alpha}} \quad (\text{A.5})$$

will be

$$T_v = \frac{\sum_{\alpha} \kappa_{\alpha} \mu_{\alpha} T_{\alpha}}{\sum_{\alpha} \kappa_{\alpha} \mu_{\alpha}}. \quad (\text{A.6})$$

This is exactly the scheme adopted in CAVEAT to evaluate the vertex temperatures T_v .

b. 2D interpolation

In CAVEAT the 2D interpolation for calculating the vertex temperatures is based on a local bilinear interpolation on a mathematical ξ, η plane. Let us assume that a rectangle, made up by four cell centers $\vec{x}_{c\alpha}$ surrounding mesh vertex (i, j) , is mapped onto a square $-1 \leq \xi \leq +1, -1 \leq \eta \leq +1$ on a mathematical plane of “natural coordinates” ξ and η (see Fig. A.2). Then an obvious bilinear interpolation

$$\begin{aligned} T(\xi, \eta) = & T_{(+1,+1)} \frac{1}{4}(1 + \xi)(1 + \eta) + T_{(-1,+1)} \frac{1}{4}(1 - \xi)(1 + \eta) + \\ & T_{(-1,-1)} \frac{1}{4}(1 - \xi)(1 - \eta) + T_{(+1,-1)} \frac{1}{4}(1 + \xi)(1 - \eta) \end{aligned} \quad (\text{A.7})$$

can be written for any function $T(x, y)$ (in this subsection we use the notation $x_1 = x, x_2 = y$), based on four values $T_{(\pm 1, \pm 1)}$ at the corresponding corners $\xi = \pm 1, \eta = \pm 1$ of the “natural” square.

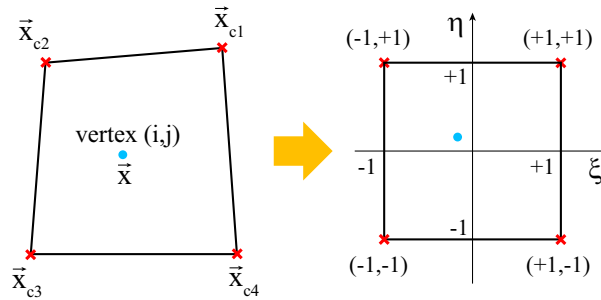


FIG. A.2: 2D mapping of a rectangular cell (i, j) in physical space onto a “natural” square in mathematical ξ, η space.

It is assumed that the mapping of physical coordinates x, y onto the ξ, η plane is performed by the same bilinear interpolation (A.7) with $T(\xi, \eta)$ replaced by either $x(\xi, \eta)$ or $y(\xi, \eta)$.

Then the values of ξ and η , corresponding to the vertex point $\vec{x}_{ij} = \vec{x} = \{x; y\}$, can be found from the following system of equations,

$$\begin{aligned} x = & \frac{1}{4}(x_{c1} + x_{c2} + x_{c3} + x_{c4}) + \\ & \frac{1}{4}(x_{c1} - x_{c2} - x_{c3} + x_{c4})\xi + \\ & \frac{1}{4}(x_{c1} + x_{c2} - x_{c3} - x_{c4})\eta + \\ & \frac{1}{4}(x_{c1} - x_{c2} + x_{c3} - x_{c4})\xi\eta, \end{aligned} \tag{A.8}$$

$$\begin{aligned} y = & \frac{1}{4}(y_{c1} + y_{c2} + y_{c3} + y_{c4}) + \\ & \frac{1}{4}(y_{c1} - y_{c2} - y_{c3} + y_{c4})\xi + \\ & \frac{1}{4}(y_{c1} + y_{c2} - y_{c3} - y_{c4})\eta + \\ & \frac{1}{4}(y_{c1} - y_{c2} + y_{c3} - y_{c4})\xi\eta, \end{aligned} \tag{A.9}$$

which is easily reduced to a quadratic equation. Here we used the CAVEAT convention for numbering the cell centers around vertex (i, j) : center 1 is the center of cell (i, j) , center 2 is the center of cell $(i - 1, j)$, ... (see Fig. A.2).

Finally, taking into account correction for possible strong variations of the conduction coefficient κ , we obtain the following interpolation formula for the vertex temperature T_v :

$$T_v = \frac{\kappa_1 T_1 (1 + \xi)(1 + \eta) + \kappa_2 T_2 (1 - \xi)(1 + \eta) + \kappa_3 T_3 (1 - \xi)(1 - \eta) + \kappa_4 T_4 (1 + \xi)(1 - \eta)}{\kappa_1 (1 + \xi)(1 + \eta) + \kappa_2 (1 - \xi)(1 + \eta) + \kappa_3 (1 - \xi)(1 - \eta) + \kappa_4 (1 + \xi)(1 - \eta)}. \tag{A.10}$$

2. Face-centered heat flux

Consider a single mesh cell (i, j) , as it is shown in Figs. 3.1 and 5.1. For brevity, in this section we omit the indices (i, j) for all quantities. To implement the SSI algorithm for heat conduction, we need explicit expressions for the total heat fluxes H_1 and H_2 [erg s⁻¹] through the two faces 1 and 2 of this cell, which emerge from the vertex (i, j) . Here we combine the formulae for both faces by using universal vector notations.

Let

$$\vec{\lambda}_v = \vec{x}_+ - \vec{x}, \quad \vec{\lambda}_c = \vec{x}_c - \vec{x}_{c-} \tag{A.11}$$

be the two vectors connecting, respectively, the two vertices along the considered face m , and the cell centers across this same face (see Fig. A.3). Vector $\vec{\lambda}_v$ starts at vertex (i, j) ; vector $\vec{\lambda}_c$ ends at cell center (i, j) . Let further T , T_- , T_v , T_{v+} be, respectively, the values of temperature at the relevant cell centers \vec{x}_c , \vec{x}_{c-} , and the relevant vertices \vec{x} , \vec{x}_+ ; see Fig. A.3.

To calculate the flux

$$H = -\kappa_f \left[(\nabla T)_f \cdot \vec{n}_f \right] |\vec{\lambda}_v| R_f, \tag{A.12}$$

given by Eq. (5.2), we have to evaluate the face-centered conduction coefficient κ_f , and the face-centered temperature gradient

$$\vec{g} = (\nabla T)_f. \tag{A.13}$$

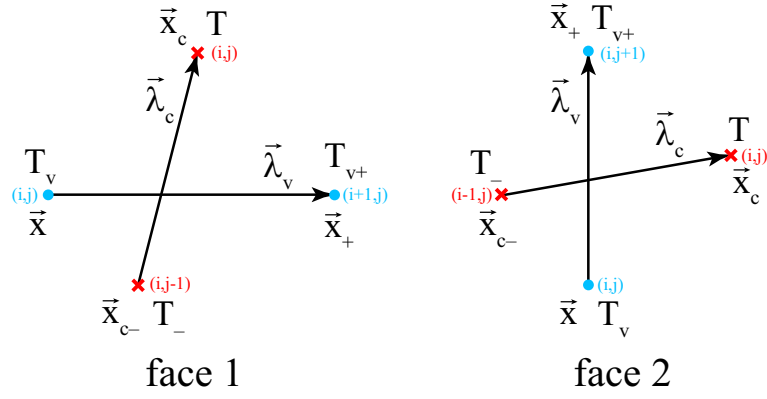


FIG. A.3: Vector scheme for temperature gradient evaluation.

The components of the inward-oriented unit normal \vec{n}_f (as defined in Fig. 3.1) are given by

$$\begin{aligned}\vec{n}_f &= \{-\lambda_{v,2}; \lambda_{v,1}\} \frac{\omega_m}{|\vec{\lambda}_v|}, \text{ face 1,} \\ \vec{n}_f &= \{\lambda_{v,2}; -\lambda_{v,1}\} \frac{\omega_m}{|\vec{\lambda}_v|}, \text{ face 2,}\end{aligned}\tag{A.14}$$

where

$$\omega_m = \begin{cases} +1, & \text{right-handed mesh,} \\ -1, & \text{left-handed mesh.} \end{cases}\tag{A.15}$$

The face-centered radius R_f is given by

$$R_f = \frac{1}{2}(R + R_+),\tag{A.16}$$

where R and R_+ are, respectively, the radii of vertices \vec{x} and \vec{x}_+ .

The gradient \vec{g} is calculated by solving the system of two linear equations

$$\begin{aligned}\vec{g} \cdot \vec{\lambda}_v &= T_{v+} - T_v, \\ \vec{g} \cdot \vec{\lambda}_c &= T - T_-, \end{aligned}\tag{A.17}$$

the mathematical meaning of which is clear from Fig. A.3. The solution of (A.17) is given by

$$\begin{aligned}g_1 &= J^{-1} [\lambda_{c,2}(T_{v+} - T_v) - \lambda_{v,2}(T - T_-)], \\ -g_2 &= J^{-1} [\lambda_{c,1}(T_{v+} - T_v) - \lambda_{v,1}(T - T_-)],\end{aligned}\tag{A.18}$$

where

$$J = \vec{\lambda}_v \otimes \vec{\lambda}_c = \lambda_{v,1}\lambda_{c,2} - \lambda_{v,2}\lambda_{c,1} = \begin{cases} +|J|\omega_m, & \text{face 1,} \\ -|J|\omega_m, & \text{face 2,} \end{cases}\tag{A.19}$$

and symbol \otimes denotes a vector product.

Finally, after we substitute Eqs. (A.14) and (A.18) into Eq. (A.12), we obtain

$$H = \frac{\kappa_f R_f}{|J|} \left[\left(\vec{\lambda}_v \cdot \vec{\lambda}_c \right) (T_{v+} - T_v) - |\vec{\lambda}_v|^2 (T - T_-) \right]. \quad (\text{A.20})$$

This is the basic expression which enables us to calculate both the explicit fluxes H_{ijm} and the coefficients a_{ijm} , b_{ijm} required in the SSI method (see section 4 above).

3. Face-centered conduction coefficient

In the original CAVEAT version the face-centered conduction coefficient κ_f is evaluated as an average of the two adjoining cell-centered values,

$$\kappa_{f,ij1} = \kappa_{ij} \frac{A_{\Delta,ij1}^-}{A_{\Delta,ij1}^- + A_{\Delta,ij1}^+} + \kappa_{i,j-1} \frac{A_{\Delta,ij1}^+}{A_{\Delta,ij1}^- + A_{\Delta,ij1}^+}, \quad (\text{A.21})$$

$$\kappa_{f,ij2} = \kappa_{ij} \frac{A_{\Delta,ij2}^-}{A_{\Delta,ij2}^- + A_{\Delta,ij2}^+} + \kappa_{i-1,j} \frac{A_{\Delta,ij2}^+}{A_{\Delta,ij2}^- + A_{\Delta,ij2}^+}. \quad (\text{A.22})$$

The corresponding weights are proportional to the areas $A_{\Delta,ijm}^\pm$ of the two adjacent triangles — i.e. to normal distances from the adjacent cell centers to the face (i, j, m) ; see Fig. 5.3. The idea is that if, for example, a cell center (i, j) lies very close to face (i, j, m) (i.e. $A_{\Delta,ijm}^+ \rightarrow 0$), then we must have $\kappa_{f,ijm} = \kappa_{ij}$. Note that area values $A_{\Delta,ijm}^\pm$, calculated as vector products, can be negative. An additional constraint is imposed that the weight coefficients in Eqs. (A.21) and (A.22) should lie between 0 and 1.

-
- [1] P. Woodward and Ph.Colella, *J. Comp. Physics* **54** (1984) 115.
 - [2] P. Woodward and Ph.Colella, *J. Comp. Physics* **54** (1984) 174.
 - [3] R.B. Pember, R.W. Anderson, *Comparison of Direct Eulerian Godunov and Lagrange Plus Remap, Artificial Viscosity Schemes*, UCRL-JC-143206 (Livermore, 2001).
 - [4] R. Ramis and J. Meyer-ter-Vehn, MPQ report 174, 1992; <http://server.faiia.upm.es/multi>.
 - [5] M.M. Basko, J. Maruhn, and T. Schlegel, *Phys. Plasmas*, **9** (2002) 1348.
 - [6] M.M. Basko, T. Schlegel, and J. Maruhn, *Phys. Plasmas*, **11** (2004) 1577.
 - [7] F. L. Addessio, J. R. Baumgardner, J. K. Dukowicz, N. L. Johnson, B. A. Kashiwa, R. M. Rauenzahn, and C. Zemach, *CAVEAT: A Computer Code for Fluid Dynamics Problems With Large Distortion and Internal Slip*, LA-10613-MS, Rev. 1, UC-905 (Los Alamos, 1992).
 - [8] A.I. Shestakov, J.L. Milovich, and M.K. Prasad, *J. Comp. Physics* **170** (2001) 81.
 - [9] E. Livne and A. Glasner, *J. Comp. Physics* **58** (1985) 59.
 - [10] Ya. B. Zel'dovich and Yu. P. Raizer, *Physics of Shock-Waves and High-Temperature Hydrodynamic Phenomena. Vol. II.* (Academic Press, New York, 1967.)