Lecture Notes for the Course on
NUMERICAL METHODS FOR NUCLEAR REACTORS

Prof. WALTER AMBROSINI

University of Pisa, Italy

Unit-6 - Neutron Transport Theory Fundamentals and Solution Methods – Part 2

NOTICE: These notes were prepared by Prof. Ambrosini mainly **on the basis of the material adopted by Prof. Bruno Montagnini for the lectures he held up to years ago**, when he left to Prof. Ambrosini the charge of the course held for the Degree in Nuclear Engineering at the University of Pisa. This material is freely distributed to Course attendees or to anyone else requesting it. It has not the worth of a textbook and it is not intended to be an official publication. It was conceived as the notes that the teacher himself would take of his own lectures in the paradoxical case he could be both teacher and student at the same time (sometimes space and time stretch and fold in strange ways). It is also used as slides to be projected during lectures to assure a minimum of uniform, constant quality lecturing, regardless of the teacher's good and bad days. As such, the material contains reference to classical textbooks and material whose direct reading is warmly recommended to students for a more accurate understanding. In the attempt to make these notes as original as feasible and reasonable, considering their purely educational purpose, most of the material has been completely re-interpreted in the teacher's own view and personal preferences about notation. In this effort, errors in details may have been introduced which will be promptly corrected in further versions after discovery. Requests of clarification, suggestions, complaints or even sharp judgements in relation to this material can be directly addressed to Prof. Ambrosini at the e-mail address: walter.ambrosini@ing.unipi.it

SPATIAL ASYMPTOTIC APPROXIMATIONS FOR THE SPHERICAL HARMONICS METHOD

The plane case with dependence on energy

- **Taking into account the dependence on energy, the transport equation for the plane case is:**

$$\mu\frac{\partial\phi(x,E,\mu)}{\partial x}+\Sigma_t(x,E)\phi(x,E,\mu)=$$

$$=\sum_{l=0}^{\infty}\frac{2l+1}{4\pi}\int dE'\int_0^{2\pi}d\varphi'\int_{-1}^{1}\Sigma_{sl}(x,E'\to E)P_l(\mu_0)\phi(x,E',\mu')d\mu'+S(x,E,\mu)$$

- **Making use of mathematical developments similar to the ones adopted for the monokinetic case and assuming**

$$\phi(x,E,\mu)=\sum_{m=0}^{\infty}\frac{2m+1}{4\pi}\phi_m(x,E)P_m(\mu)\ \textbf{with}\ \phi_m(x,E)=2\pi\int_{-1}^{1}\phi(x,E,\mu)P_m(\mu)d\mu$$

$$S(x,E,\mu)=\sum_{m=0}^{\infty}\frac{2m+1}{4\pi}S_m(x,E)P_m(\mu)\ \textbf{with}\ S_m(x,E)=2\pi\int_{-1}^{1}S(x,E,\mu)P_m(\mu)d\mu$$

it is possible to reach the energy dependent spherical harmonics form (we omit the demonstration)

$$\frac{l}{2l+1}\frac{\partial\phi_{l-1}(x,E)}{\partial x}+\frac{l+1}{2l+1}\frac{\partial\phi_{l+1}(x,E)}{\partial x}+\Sigma_t(x,E)\phi_l(x,E)$$

$$=\int\Sigma_{sl}(x,E'\to E)\phi_l(x,E')dE'+S_l(x,E)\qquad(l=0,1,...)$$

being a system of infinite integrodifferential equations in $\phi_l(x,E)$

- **The P_N approximation consists in truncating the series at the $(N+1)$-th term obtaining $N+1$ equations ($l=0,1,...,N$)**

Asympotitc spatial dependence for P_N and the B_N approximations

- **In the case of homogeneous regions it is possible to adopt a procedure different from the spatial discretization, based on a simple *asymptotic approximation***
- **In the case of calculations aimed to provide the fine group energy spectrum, a spatial trend of the flux of exponential type (for non multiplying media) or sinusoidal one (for multiplying media) may roughly represent the effect of leakages**

- **Therefore, for a multiplying medium an imaginary exponential trend is accepted. The imaginary exponential is more convenient from a mathematical point ot view than a sinusoidal one, just taking care to consider only the real part of both the flux and the source (assumed to be isotropic)**

$$\phi(x,\mu,E)=\varphi(\mu,E)e^{-iBx} \qquad S(x,\mu,E)=\frac{1}{4\pi}S_0(E)e^{-iBx}$$

- **As a consequence, for the coefficients of the Legendre polynomials it is**

$$\varphi_n(x,E)=\varphi_n(E)e^{-iBx}$$

- **With these definitions, substituting into the relation**

$$\mu\frac{\partial\phi(x,E,\mu)}{\partial x}+\Sigma_t(x,E)\phi(x,E,\mu)$$

$$=\sum_{l=0}^{\infty}\frac{2l+1}{4\pi}P_l(\mu)\int\Sigma_{sl}(x,E'\to E)\phi_l(x,E')dE'+S(x,E,\mu)$$

and dividing both sides by the common factor e^{-iBx}, it is obtained

$$[-iB\mu+\Sigma_t(E)]\varphi(\mu,E)=\sum_{n=0}^{\infty}\frac{2n+1}{4\pi}P_n(\mu)\int\Sigma_{sn}(E'\to E)\varphi_n(E')dE'+\frac{1}{4\pi}S_0(E)$$

On this basis, we can now proceed in <u>two different ways</u>:

1. <u>P_N equations in asymptotic form</u>

We can substitute the expression of the flux in terms of Legendre polynomials at the left hand side, by the relation:

$$\varphi(\mu,E)=\sum_{n=0}^{\infty}\frac{2n+1}{4\pi}\varphi_n(E)P_n(\mu)$$

Making use for $\mu P_n(\mu)$ of the equavalent expression obtained in terms of Legendre polynomials $\frac{n+1}{2n+1}P_{n+1}(\mu)+\frac{n}{2n+1}P_{n-1}(\mu)$, multiplying both side by $P_l(\mu)$ and integrating on $-1\le\mu\le1$, it is found:

$$-iB\left[\frac{l}{2l+1}\varphi_{l-1}(E)+\frac{l+1}{2l+1}\varphi_{l+1}(E)\right]+\Sigma_t(E)\varphi_l(E)=\int\Sigma_{sl}(E'\to E)\varphi_l(E')dE'+\delta_{0l}S_0(E)$$

$$(l=0,1,\ldots,N)$$

which represent the <u>P_N equations in asymptotic form</u>.

2. B_N Equations

As a variant of the above, before including the expression of the flux in terms of Legendre polynomials and of performing the "scalar product" by $P_l(\mu)$, we divide both sides by $\Sigma_t(E)-iB\mu$:

$$\varphi(\mu,E)=\frac{1}{\Sigma_t(E)-iB\mu}\sum_{n=0}^{\infty}\frac{2n+1}{4\pi}P_n(\mu)\int\Sigma_{sn}(E'\to E)\varphi_n(E')dE'+\frac{1}{4\pi}\frac{S_0(E)}{\Sigma_t(E)-iB\mu}$$

Then we multiply by $P_l(\mu)$ and we integrate from -1 to 1:

$$\int_{-1}^{1}\varphi(\mu,E)P_l(\mu)d\mu=\frac{\varphi_l(E)}{2\pi}=$$

$$=\sum_{n=0}^{\infty}\frac{2n+1}{4\pi}\int_{-1}^{1}\frac{P_n(\mu)P_l(\mu)}{\Sigma_t(E)-iB\mu}d\mu\int\Sigma_{sn}(E'\to E)\varphi_n(E')dE'+\frac{S_0(E)}{4\pi}\int_{-1}^{1}\frac{P_l(\mu)}{\Sigma_t(E)-iB\mu}d\mu$$

and we multiply both sides by $2\pi\Sigma_t(E)$

$$\Sigma_t(E)\varphi_l(E)=\sum_{n=0}^{\infty}\frac{2n+1}{2}\int_{-1}^{1}\frac{P_n(\mu)P_l(\mu)}{1-i\dfrac{B}{\Sigma_t(E)}\mu}d\mu\int\Sigma_{sn}(E'\to E)\varphi_n(E')dE'+\frac{S_0(E)}{2}\int_{-1}^{1}\frac{P_l(\mu)P_0(\mu)}{1-i\dfrac{B}{\Sigma_t(E)}\mu}d\mu$$

We now introduce the coefficients (new functions to be calculated on the basis of Legendre polynomials):

$$A_{l,n}(z)=\frac{1}{2}\int_{-1}^{1}\frac{P_l(\mu)P_n(\mu)}{1-iz\mu}d\mu$$

obtaining

$$\Sigma_t(E)\varphi_l(E)=\sum_{n=0}^{\infty}(2n+1)A_{l,n}\left(\frac{B}{\Sigma_t(E)}\right)\int\Sigma_{sn}(E'\to E)\varphi_n(E')dE'+A_{l,0}\left(\frac{B}{\Sigma_t(E)}\right)S_0(E)$$

The functions $A_{l,n}$ satisfy the recurrence condition

$$\frac{1}{iz}(2n+1)A_{l,n}(z)-(n+1)A_{l,n+1}(z)-lA_{l,n-1}(z)=\frac{\delta_{l,n}}{iz}$$

and it is

$$A_{n,l}=A_{l,n}\quad\textbf{e}\qquad A_{0,0}(z)=\frac{1}{z}\operatorname{arctg}z\approx1-\frac{z^2}{3}+\frac{z^4}{5}-\frac{z^6}{7}+\ldots\ per\ z\approx0$$

$$A_{0,1}(z)=A_{1,0}(z)=\frac{1}{iz}\left[A_{0,0}(z)-1\right]\qquad A_{1,1}(z)=\frac{1}{iz}A_{0,1}(z)$$

The B_N approximation consists again in truncating at $l=N$ the system of infinite integral equations thus obtained. It can be noted that:

- **this occurs automatically, putting $\Sigma_{sn}(E'\to E)=0$ for $n>N$**
  - **the B_N equations converge more rapidly than the corresponding P_N; for instance, in the case of isotropic**

scattering, an "exact" expression is obtained for φ_0 and also the higher order coefficients can be obtained exactly;

- **it is foreseeable that also for anisotropic scattering the process converges more rapidly, as shown below.**

For $N=0$ (B_0 approximation) for $l=0$ we have

$$\Sigma_t(E)\varphi_0(E)=A_{0,0}\left(\frac{B}{\Sigma_t(E)}\right)\left[\int \Sigma_{s0}(E'\to E)\varphi_0(E')dE'+S_0(E)\right]$$

and this equation is "exact" (no need of truncation of higher order terms!); in order to get the angular flux in addition to φ_0, it is possible to write

$$\varphi(\mu,E)=\frac{1}{\Sigma_t(E)-iB\mu}\left[\frac{1}{4\pi}\int \Sigma_{s0}(E'\to E)\varphi_0(E')dE'+\frac{S_0(E)}{4\pi}\right]$$

As an alternative, it is possible to obtain all the needed $\varphi_l(E)$ from the equation

$$\Sigma_t(E)\varphi_l(E)=\sum_{n=0}^{\infty}(2n+1)A_{l,n}\left(\frac{B}{\Sigma_t(E)}\right)\int \Sigma_{sn}(E'\to E)\varphi_n(E')dE'+A_{l,0}\left(\frac{B}{\Sigma_t(E)}\right)S_0(E)$$

whose RHS involves only φ_0 and S_0.

For $N=1$ (B_1 approximation) with some passages, the following system of two equations in two unknowns is obtained

$$\Sigma_t(E)\varphi_0(E)-iB\varphi_1(E)=\int \Sigma_{s0}(E'\to E)\varphi_0(E')dE'+S_0(E)$$

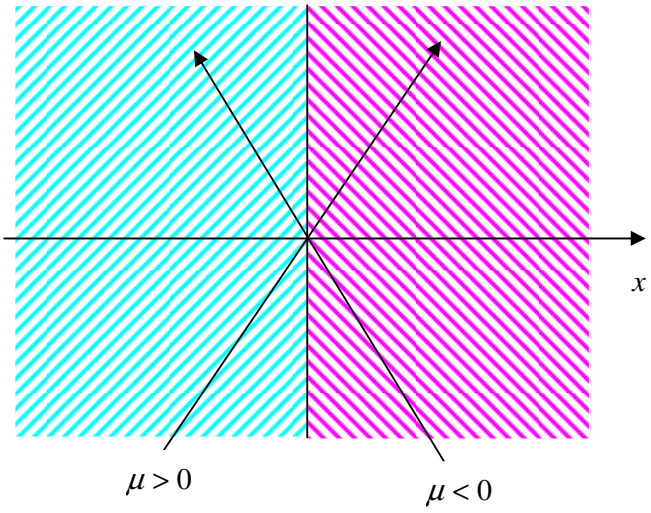
$$g\left(\frac{B}{\Sigma_t(E)}\right)\Sigma_t(E)\varphi_1(E)-\frac{iB}{3}\varphi_0(E)=\int \Sigma_{s1}(E'\to E)\varphi_1(E')dE'$$

where

$$g(z)=\frac{z^2}{3}\frac{A_{0,0}(z)}{1-A_{0,0}(z)} \qquad (\approx 1-\frac{z^2}{3} \text{ per } z\approx 0)$$

As it can be noted, an advantage with respect to the P_N equations is that the B_N equations do not refer to higher order components of the flux (no need for truncating! $\Rightarrow$ faster convergence)

DP_N APPROXIMATIONS



- **The expression of the angular flux in terms of Legendre polynomials is inadequate for dealing with the discontinuity at a planar surface (across $\mu = 0$)**
- **In fact, the neutrons contributing to the angular flux for $\mu > 0$ are generated in one region, while those contributing to the flux for $\mu < 0$ come from the other, and the two regions can be quite different in terms of sources and properties**
- **This is not true for a curved surface. However, the presence of such a discontinuity is very evident in the case of an interface with the void)**

$$\phi(x_s,\mu) = 0 \qquad \mu < 0 \quad \textbf{or} \quad \phi(x_s,\mu) = 0 \qquad \mu > 0$$

depending on where the interface is located

- **In order to overcome this difficulty, J.J. Yvon proposed to adopt different expansions of the angular flux in the two intervals $-1 \le \mu \le 0$ and $0 \le \mu \le 1$**
- **Considering for the sake of simplicity the monokinetic case, it is:**

$$\phi(x,\mu) = \sum_{n=0}^{N} (2n+1)\left[\phi_n^+(x) P_n^+(2\mu - 1) + \phi_n^-(x) P_n^-(2\mu + 1)\right]$$

where

$$P_n^+(2\mu-1) = \begin{cases} P_n(2\mu-1) & \mu \ge 0 \\ 0 & \mu < 0 \end{cases} \qquad P_n^-(2\mu+1) = \begin{cases} P_n(2\mu+1) & \mu < 0 \\ 0 & \mu \ge 0 \end{cases}$$

- **Substituting this expression in the steady state equation of neutron transport for the monokineitc case and for a generally anisotropic scattering (without source)**

$$\mu \frac{\partial \phi(x,\mu)}{\partial x} + \Sigma_t(x)\phi(x,\mu) = \sum_{l=0}^{\infty} \frac{2l+1}{2} \Sigma_{sl}(x) P_l(\mu) \int_{-1}^{1} \phi(x,\mu') P_l(\mu') d\mu'$$

and multiplying both sides by $P_m^+(2\mu - 1)$ o $P_m^-(2\mu + 1)$ then integrating on $-1 \le \mu \le 1$, it is:

$$\frac{m}{2m+1}\frac{d\phi_{m-1}^{\pm}(x)}{dx}+\frac{m+1}{2m+1}\frac{d\phi_{m+1}^{\pm}(x)}{dx}\pm\frac{d\phi_{m}^{\pm}(x)}{dx}+2\Sigma_t(x)\phi_m^{\pm}(x)$$
$$=\sum_{l=0}^{\infty}(2l+1)p_{lm}^{\pm}\Sigma_{sl}(x)\sum_{n=0}^{N}(2n+1)\left[p_{ln}^{+}\phi_n^{+}(x)+p_{ln}^{-}\phi_n^{-}(x)\right]$$

where $\pm$ identifies the + or – sign suggesting that the related equation has been obtained by multiplying by $P_m^{+}(2\mu-1)$ or by $P_m^{-}(2\mu+1)$ and where it is assumed

$$p_{lm}^{+}=\int_{-1}^{1}P_l(\mu)P_m^{+}(2\mu-1)d\mu \qquad p_{lm}^{-}=\int_{-1}^{1}P_l(\mu)P_m^{-}(2\mu+1)d\mu$$

- **When the series defining the angular flux is truncated at the N-th term and considering only the first $N+1$ equations, the *double-P_N* or *DP_N approximation* is obtained**
- **In some cases of interest, this approximation is much better than the corresponding P_N one. In particular, putting**

$$\phi_n^{+}(0)=0 \qquad \textbf{and} \qquad \phi_n^{-}(a)=0 \quad (n=0,1,...)$$

It is easier to satisfy in an appropriate way the boundary conditions for the interface with the void in the case of a layer (in $x=0$ an $x=a$)

- **The DP_N method requires that in every point it is possible to identify directions pointing outwards and inwards, something only possible for 1D cases**
- **However, it must be mentioned that the P_N method can be extended to multidimensional cases only with great mathematical difficulty**

THE INTEGRAL EQUATION FOR 1D AND 2D PROBLEMS

1D Geometry

- **Considering an isolated slab included in the interval $0 \le x \le a$ <u>with isotropic scattering and indipendent source</u>, in Cartesian coordinates the transport equation in integral form**

$$\phi(\vec{r},E)=\int_V \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2}\left[\int_0^\infty \Sigma_{s0}(\vec{r}',E'\to E)\phi(\vec{r}',E')dE'+S_0(\vec{r}',E)\right]dV'$$

becomes

$$\phi(x,E)=\int_V \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2}\left[\int_0^\infty \Sigma_{s0}(x',E'\to E)\phi(x',E')dE'+S_0(x',E)\right]dx'dy'dz'$$

or

$$\phi(x,E)=\int_0^a dx'\left[\int_0^\infty \Sigma_{s0}(x',E'\to E)\phi(x',E')dE'+S_0(x',E)\right]\int_{-\infty}^{\infty}dy'\int_{-\infty}^{\infty}\frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2}dz'$$

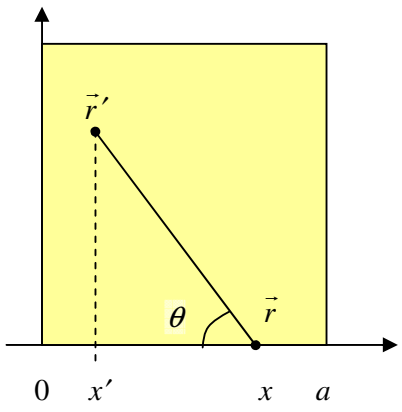
where the integration along the coordinates other than *x* is isolated at the right, to be made first as as afucntion of any *x*

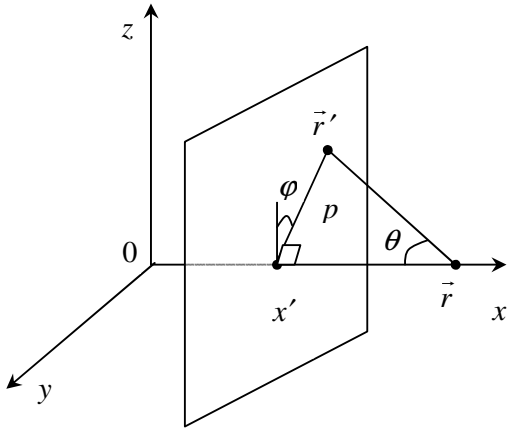
- **It is now convenient to change the integration variables, so that we can take full profit of the 1D characteristics of the problem**
- **The reference frame is selected such that**

$$\vec{r}=\{x,0,0\} \qquad\qquad \vec{r}'=\{x',y',z'\}$$

then introducing polar coordinates in the plane $y'z'$, with azimuth φ and vector radius p

$$p=\sqrt{y'^2+z'^2}$$





- **It is therefore**

$$\int_{-\infty}^{\infty} dy' \int_{-\infty}^{\infty} \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2} dz' = \int_0^{2\pi} d\varphi \int_0^{\infty} \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi\left[(x-x')^2+p^2\right]} pdp$$

- **Now it is therefore convenient to substitute to p the variable t defined as:**

$$t \equiv \frac{1}{|\cos\theta|} = \frac{|\vec{r}-\vec{r}'|}{|x-x'|} = \frac{\sqrt{(x-x')^2+p^2}}{|x-x'|} \qquad (1 \le t < \infty)$$

obtaining

$$dt = \frac{1}{|x-x'|} \frac{pdp}{\sqrt{(x-x')^2+p^2}} \quad \Rightarrow \quad \frac{dt}{t} = \frac{pdp}{(x-x')^2+p^2} = \frac{pdp}{|\vec{r}-\vec{r}'|^2}$$

and then

$$p\,dp\,d\varphi = |\vec{r}-\vec{r}'|^2 \frac{dt}{t} d\varphi$$

- **We have therefore:**

$$\int_{-\infty}^{\infty} dy' \int_{-\infty}^{\infty} \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2} dz' = \int_0^{2\pi} d\varphi \int_0^{\infty} \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi\left[(x-x')^2+p^2\right]} pdp = \frac{1}{2}\int_1^{\infty} e^{-\tau(x,x',E)t} \frac{dt}{t}$$

where it can be noted that <u>we used the "obvious" relation</u>:

$$\tau(\vec{r},\vec{r}',E) = \frac{\tau(x,x',E)}{|\cos\theta|} = \tau(x,x',E)t$$

In fact, we recognise that when the neutron trajectory is inclined with respect to the x axis, this results just in an increase by a factor $1/|\cos\theta|$ in the number of mean free paths involved in their flight

- **So defining the <u>*exponential integral functions*</u>**

$$E_n(z) = \int_1^{\infty} e^{-zt} \frac{dt}{t^n}$$

it is

$$\phi(x,E) = \frac{1}{2}\int_0^a E_1[\tau(x,x',E)]\left[\int_0^{\infty} \Sigma_{s0}(x',E'\to E)\phi(x',E')dE' + S_0(x',E)\right]dx'$$

- **In the monokinetic case it is:**

$$\phi(x) = \frac{1}{2}\int_0^a E_1[\tau(x,x')][\Sigma_{s0}(x')\phi(x') + S_0(x')]dx'$$

- **When the layer $[0,a]$ is homogeneous (same material everywhere), it is simply $\tau(x,x') = \Sigma_t|x-x'|$ and therefore**

$$\phi(x) = \frac{1}{2}\int_0^a E_1[\Sigma_t|x-x'|][\Sigma_{s0}(x')\phi(x') + S_0(x')]dx'$$

- **If we choose the meen free path as length scale (i.e., substituting to x the coordinate $\tau = \Sigma_t x$) it is**

$$\phi(\tau) = \frac{1}{2\Sigma_t}\int_0^{\Sigma_t a} E_1\left(|\tau - \tau'|\right)\left[\Sigma_{s0}(\tau')\phi(\tau') + S_0(\tau')\right]d\tau'$$

- **This relation can be reached also for a non-homogeneous layer. Putting:**

$$x \to \tau(x) = \int_0^x \Sigma_t(x'')dx'' \qquad d\tau = \Sigma_t(x)dx$$

and, expressing the flux as a function of τ in place of x, it is

$$\phi(\tau) = \frac{1}{2}\int_0^{\tau_a} E_1\left(|\tau - \tau'|\right)\left[c(\tau')\phi(\tau') + s_0(\tau')\right]d\tau'$$

where

$$c = \frac{\Sigma_{s0}}{\Sigma_t} \qquad \textbf{and} \qquad s_0 = \frac{S_0}{\Sigma_t}$$

That is coincident with reation obtained in the case of a homogenoeus layer by substituting $\Sigma_t a$ with the more general form $\tau_a = \int_0^a \Sigma_t(x)dx$.

2D Geometry

- **We consder a "geenralized cylinder" (not necessarily circular) infinite in height and isolated having a volume V having as intersection with the plane $z = 0$ the general surface A_0**
- **We also assume that <u>the scattering and the independent source are isotropic</u>**
- **If the variables in the problem depend only on x and y, the integral transport equation becomes**

$$\phi(x,y,E) = \int_V \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2}\left[\int_0^\infty \Sigma_{s0}(x',y',E'\to E)\phi(x',y',E')dE' + S_0(x',y',E)\right]dx'dy'dz'$$

or

$$\phi(x,y,E) = \int_{A_0} dx'dy'\left[\int_0^\infty \Sigma_{s0}(x',y',E'\to E)\phi(x',y',E')dE' + S_0(x',y',E)\right]\int_{-\infty}^{\infty}\frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2}dz'$$

where we now isolated in the RHS the integration over *z*

- **Therefore, also in the 2D case an appropriate choice of the coordinate helps in obtaining a final compact form; it is:**

$$\vec{r} = \vec{r}_0 \equiv \{x, y, 0\} \qquad \textbf{and} \quad \vec{r}' \equiv \{x', y', z'\}$$

- **Putting**

$$R_0 = \sqrt{(x-x')^2 + (y-y')^2} \qquad R(R_0, z') = \sqrt{R_0^2 + z'^2} = \frac{R_0}{\cos\theta}$$

we have

$$z' = R_0 tg\theta \quad \Rightarrow \quad dz' = \frac{R_0 d\theta}{\cos^2\theta}$$

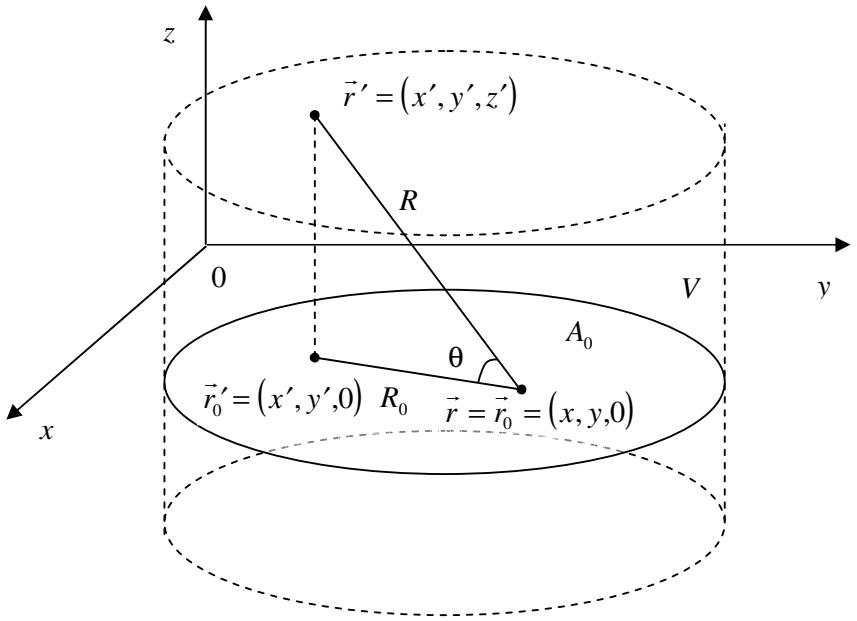
- **For the particular case of the homogeneous generalised cylinder, it is** $\Sigma_t = cost.$ **and then**

$$\int_{-\infty}^{\infty} \frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2} dz' = \int_{-\infty}^{\infty} \frac{e^{-\Sigma_t\sqrt{(x-x')^2+(y-y')^2+z'^2}}}{4\pi\left[(x-x')^2+(y-y')^2+z'^2\right]} dz' = \int_{-\infty}^{\infty} \frac{e^{-\Sigma_t R(R_0,z')}}{4\pi R^2(R_0,z')} dz'$$

$$= \int_{-\pi/2}^{\pi/2} \frac{e^{-\Sigma_t R_0/\cos\theta}}{4\pi(R_0/\cos\theta)^2} \frac{R_0 d\theta}{\cos^2\theta} = \frac{2}{4\pi R_0}\int_0^{\pi/2} e^{-\Sigma_t R_0/\cos\theta} d\theta = \frac{1}{2\pi R_0} Ki_1(\Sigma_t R_0)$$

where the *Bickley-Naylor functions* were used. They are defined as:

$$Ki_n(x) = \int_0^{\pi/2} e^{-x/\cos\theta}(\cos\theta)^{n-1} d\theta = \int_1^{\infty} e^{-xt} \frac{dt}{t^n\sqrt{t^2-1}} \qquad (n = 1,2,3,...)$$



Reference geometry for the 2D case

- **In the general case in which the cylinder is non-homogeneous, it is**

$$\tau(\vec{r},\vec{r}',E)=\frac{\tau(\vec{r}_0,\vec{r}_0',E)}{\cos\theta}$$

and then

$$\int_{-\infty}^{\infty}\frac{e^{-\tau(\vec{r},\vec{r}',E)}}{4\pi|\vec{r}-\vec{r}'|^2}dz'=\frac{1}{2\pi}\int_0^{\pi/2}\frac{e^{-\tau(\vec{r}_0,\vec{r}_0',E)/\cos\theta}}{\left(|\vec{r}_0-\vec{r}_0'|/\cos\theta\right)^2}\frac{|\vec{r}_0-\vec{r}_0'|d\theta}{\cos^2\theta}$$

$$=\frac{1}{2\pi|\vec{r}_0-\vec{r}_0'|}\int_0^{\pi/2}e^{-\tau(\vec{r}_0,\vec{r}_0',E)/\cos\theta}d\theta=\frac{1}{2\pi|\vec{r}_0-\vec{r}_0'|}Ki_1\left[\tau(\vec{r}_0,\vec{r}_0',E)\right]$$

- **It can be then concluded that**

$$\phi(\vec{r}_0,E)=\int_{A_0}\frac{Ki_1\left[\tau(\vec{r}_0,\vec{r}_0',E)\right]}{2\pi|\vec{r}_0-\vec{r}_0'|}\left[\int_0^{\infty}\Sigma_{s0}(\vec{r}_0',E'\to E)\phi(\vec{r}_0',E')dE'+S_0(\vec{r}_0',E)\right]dA_0'$$

- **In the particular case of a monokinetic problem**

$$\phi(\vec{r}_0)=\int_{A_0}\frac{Ki_1\left[\tau(\vec{r}_0,\vec{r}_0')\right]}{2\pi|\vec{r}_0-\vec{r}_0'|}\left[\Sigma_{s0}(\vec{r}_0')\phi(\vec{r}_0')+S_0(\vec{r}_0')\right]dA_0'$$

- **Finally, if the cylinder is homogeneous**

$$\phi(\vec{r}_0)=\int_{A_0}\frac{Ki_1\left[\Sigma_t|\vec{r}_0-\vec{r}_0'|\right]}{2\pi|\vec{r}_0-\vec{r}_0'|}\left[\Sigma_{s0}\phi(\vec{r}_0')+S_0(\vec{r}_0')\right]dA_0'$$

FINAL CONSIDERATIONS

The above explains why in 1D and 2D codes based on the integral equations, collision probabilities involve exponential integral functions and Bickley-Naylor functions, respectively

As usual, we note that in reduced dimensionality problems *we need anyway to consider that neutrons travel in the 3D space*, integrating along the neglected dimensions in order to obtain the overall contribution of neutron sources *that are actually distributed in a 3D space*

DISCRETE ORDINATE METHOD OR "S_N" METHOD

General considerations

- **The discrete ordinate method is based on the solution of the interodifferential equation *discretised both in the space and in the angular coordinates***
- **This method represents the main technique for the solution of the integrodifferential transport equation since it allows to easily obtain a solution with any degree of approximation as a function of the available computational resources**
- **The first algorithms of these methods can be traced back to methods adopted for stellar atmospheres; the technique was then extended mainly owing to B. Carlson to nuclear energy applications**
- **The remarkable efficiency of these methods, named S_N metods, makes them to be often preferred to others**

The one-dimensional case in cartesian coordinates

Discretised equations

- **From the steady-state integro-differential equation**

$$\vec{\Omega}\cdot grad_{\vec{r}}\phi\left(\vec{r},v\vec{\Omega}\right)+\Sigma_t\left(\vec{r},v\right)\phi\left(\vec{r},v\vec{\Omega}\right)=\int\Sigma_s\left(\vec{r},v'\vec{\Omega}'\rightarrow v\vec{\Omega}\right)\phi\left(\vec{r},v'\vec{\Omega}'\right)dv'\,d\vec{\Omega}'+S\left(\vec{r},v\vec{\Omega}\right)$$

  we can firstly consider (just for simplicity) the monokinetic case with isotropic scattering

$$\vec{\Omega}\cdot grad_{\vec{r}}\phi\left(\vec{r},\vec{\Omega}\right)+\Sigma_t\left(\vec{r}\right)\phi\left(\vec{r},\vec{\Omega}\right)=\frac{\Sigma_s\left(\vec{r}\right)}{4\pi}\int\phi\left(\vec{r},\vec{\Omega}'\right)d\vec{\Omega}'+S\left(\vec{r},\vec{\Omega}\right)$$

  and we finally consider the already obtained form for 1D geometry

$$\mu\frac{\partial\phi(x,\mu)}{\partial x}+\Sigma_t(x)\phi(x,\mu)=\frac{\Sigma_s(x)}{2}\int_{-1}^{1}\phi(x,\mu')d\mu'+S(x,\mu)\qquad(^\circ)$$

- **In order to solve this equation, we define *M* discrete directions and corresponding weighting coefficients**

$$\mu_1,\mu_2,...,\mu_M\qquad\qquad w_1,w_2,...,w_M$$

- **In particular, making use of the weighting coefficients w_m (*quadrature coefficients*) it is possible to calculate in an approxomated way the integral at RHS of (°), that is:**

$$\int_{-1}^{1}\phi(x,\mu')d\mu'\approx\sum_{m=1}^{M}w_m\phi(x,\mu_m)$$

- **Owing to this discretisation of the angular coordinate Eq. (°) is transformed into the following**

$$\mu_m \frac{\partial \phi(x,\mu_m)}{\partial x} + \Sigma_t(x)\phi(x,\mu_m) = \frac{\Sigma_s(x)}{2}\sum_{n=1}^{M} w_n \phi(x,\mu_n) + S(x,\mu_m) \quad (m = 1, \ldots, M)$$

- **The choice of the weighting coefficients w_m and, then of the quadrature formulations is generally made with reference to an <u>even number of discrete ordinates</u> μ_m chosen in a <u>symmetric</u> way with respect to $\mu = 0$**
- **It is, therefore:**

$$\mu_m > 0 \qquad \mu_{M+1-m} = -\mu_m \qquad w_{M+1-m} = w_m \qquad \left(m = 1, 2, \ldots, \frac{M}{2}\right)$$

- **The reason of the choice of <u>symmetrically distributed</u> values with respect to $\mu = 0$ with <u>equal weights</u> is due to the intent to assign the same importance to particles streaming along different directions**
- **The <u>even number</u> of directions is then adopted in order to avoid the existence of a value of m such that $\mu_m = 0$; this would pose problems, since:**
  - ***the derivative term would disappear in the equation*, compelling to treat this direction in a different way with respect to the others**
  - **as already noted, *the direction characterised by $\mu = 0$ can be the one at which discontinuities may appear in flux along the angular coordinate***
- **The advantage to choose an even number of discrete ordinates also appears in particular when boundary conditions are imposed:**
  - **in the case of *pure reflection*, for instance at $x = 0$, it is:**

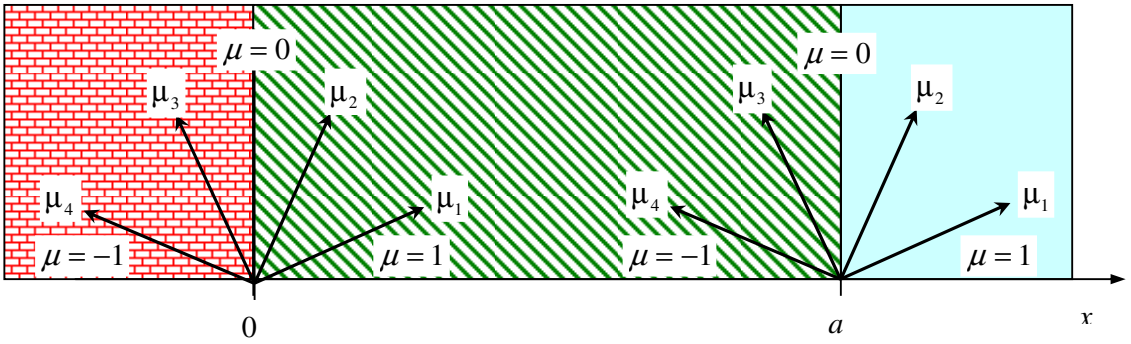
$$\phi(0,\mu_m) = \phi(0,\mu_{M+1-m}) \qquad \left(m = 1,2,\ldots,\frac{M}{2}\right)$$

  - **for *a free surface* (interface to the void) in $x = a$, instead, it is:**

$$\phi(a,\mu_m) = 0 \qquad \left(m = \frac{M}{2} + 1,\ldots,M\right)$$

- **In principle, there is anyway a considerable freedom in determining the directions**
- **A very frequent choice is the one (of *Wick-Chandrasekhar)* in which the μ_m are assigned such that they are the M zeroes of the Legendre polynomial of order M:**

$$P_M(\mu_m) = 0 \qquad (m = 1,2,\ldots,M)$$



$$\phi(0,\mu_1) = \phi(0,\mu_4)\ ;\ \phi(0,\mu_2) = \phi(0,\mu_3) \qquad\qquad \phi(a,\mu_3) = \phi(a,\mu_4) = 0$$

Pure reflection **Free surface**

Discrete ordinate in the planarcase and boundary conditions $(M = 4)$

- **Its is moreover requested that**

$$w_m > 0 \qquad (m = 1,2,\ldots,M)$$

and that the weighting is such to provide an exact integration over $-1 \le \mu \le 1$ **of all the polynomials of order up to** $M-1$**; it is therefore:**

$$\sum_{m=1}^{M} w_m \mu_m^n = \int_{-1}^{1} \mu^n d\mu = \begin{cases} \dfrac{2}{n+1}, & n\ pari \\ 0 & n\ dispari \end{cases} \qquad (n = 0,1,2,\ldots,M-1)$$

- **It is necessary to note that the previous relationships with odd** n **is identically satisfied for any set of** μ_m **and** w_m **respecting the requirements**

$$\mu_m > 0 \qquad \mu_{M+1-m} = -\mu_m \qquad w_{M+1-m} = w_m \qquad \left(m = 1,2,\ldots,\frac{M}{2}\right)$$

- **Therefore it is possible to determine the** M **independent parameters** μ_m **and** w_m $(m = 1,\ldots,M/2)$ **in order to exactly integrate all the polynomials having order** $0,2,\ldots,2M-2$ **(and also those of order** $2M-1$**, since that this is and odd number)**

- **We have therefore:**

$$\sum_{m=1}^{M} w_m P_k(\mu_m) P_l(\mu_m) = \int_{-1}^{1} P_k(\mu) P_l(\mu) d\mu = \frac{2\delta_{kl}}{2k+1} \qquad (k,l = 0,1,\ldots,M-1)$$

$N = 2$	$w_1 = w_2 = 1.000$	$\mu_1 = -\mu_2 = 0.57735$
$N = 4$	$w_2 = w_3 = 0.65215$	$\mu_2 = -\mu_3 = 0.33998$
	$w_1 = w_4 = 0.34785$	$\mu_1 = -\mu_4 = 0.86114$
$N = 6$	$w_3 = w_4 = 0.46791$	$\mu_3 = -\mu_4 = 0.23862$
	$w_2 = w_5 = 0.36076$	$\mu_2 = -\mu_5 = 0.66121$
	$w_1 = w_6 = 0.17132$	$\mu_1 = -\mu_6 = 0.93247$

Gauss-Legendre quadrature parameters (from Bell & Glasstone, 1979)

- **With these requirements μ_m and w_m are given by the *Gauss-Legendre quadrature parameters* reported in the above table for $M = 2,4,6$**
- **It is possible to show that, with this choice, the method is equivalent to the on of spherical harmonics P_N, with $N = M - 1$:**

$$S_M \equiv P_{M-1}$$

- **The numerical solution is obtained by writing the equations in the form**

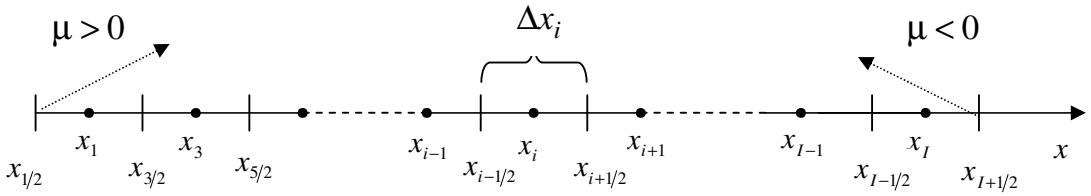
$$\mu_m \frac{\partial \phi(x,\mu_m)}{\partial x} + \Sigma_t(x)\phi(x,\mu_m) = q(x,\mu_m) \qquad (m = 1,\ldots,M)$$

and iterating on the scattering source by the scheme:

$$\mu_m \frac{\partial \phi^{[t+1]}(x,\mu_m)}{\partial x} + \Sigma_t(x)\phi^{[t+1]}(x,\mu_m) = q^{[t]}(x,\mu_m) \qquad (m = 1,\ldots,M)$$

$$q^{[t+1]}(x,\mu_m) = \frac{\Sigma_s(x)}{2}\sum_{n=1}^{M} w_n \phi^{[t+1]}(x,\mu_n) + S(x,\mu_m)$$

- **It is obviously needed also a spatial discretisation:**
  - **the interval $0 \le x \le a$ is subdivided into I subintervals with uniform properties**



Spatial discretisation for the discrete ordinate method

- **in each subinterval, the average values of angular flux, emission density and cross sections are defined**

- **the equations are then written in the discretised form:**

$$\mu_m \frac{\phi(x_{i+1/2},\mu_m)-\phi(x_{i-1/2},\mu_m)}{\Delta x_i}+\Sigma_t(x_i)\phi(x_i,\mu_m)=q(x_i,\mu_m)$$

that are rewritten using a straightforward shorthand notation for the angular flux

$$\mu_m \frac{\phi_{i+1/2,m}-\phi_{i-1/2,m}}{\Delta x_i}+\Sigma_{ti}\phi_{i,m}=q_{i,m} \qquad (i=1,...,I) \quad (m=1,...,M) \qquad (^\circ)$$

Solution algorithm

- **For any direction μ_m, the previous equations represent a system of I equations in the I unknowns of the angular flux values in the centre of each subinterval, $\phi_{i,m}$**
- **However, since in the equations also the interfacial fluxes appear, it is necessary to make use of further information in order to carry on the calculations**
- **A first information is provided by the calculation in an adjoining node, that is assumed to be already completed, or by a boundary condition**
- **In the aim to eliminate the residual unknown the so-called *diamond rule* is used**

$$\phi_{i,m}=\frac{\phi_{i+1/2,m}+\phi_{i-1/2,m}}{2}$$

- **The solution algorithm makes use of a different use of of this rule according to the sign of the direction cosine ($\mu_m>0$ or $\mu_m<0$). In particular, it is:**
  - **for $\mu_m>0$**

    in this case, the calculation proceeds <u>sweeping the subintervals from left to right</u> starting with $\phi_{1/2,m}$, to be assumed known; it is interesting to note that <u>this is also the direction of propagation of neutrons</u>; it is therefore

$$\phi_{i+1/2,m}=2\phi_{i,m}-\phi_{i-1/2,m}$$

    that, introduced into (°) provides the central flux in the form

$$\phi_{i,m} = \frac{\phi_{i-1/2,m} + \frac{\Delta x}{2\mu_m} q_{i,m}}{1 + \frac{\Delta x}{2\mu_m}\Sigma_{ti}} \qquad (\circ\circ)$$

once the central flux is known, it is therefore possible to evaluate the interfacial flux $\phi_{i+1/2,m} = 2\phi_{i,m} - \phi_{i-1/2,m}$ to be used in the calculation of the next subinterval;

- **for $\mu_m < 0$**

  unlike in the case $\mu_m > 0$, <u>the calculation proceeds from right to left</u>; it is again worth to note that <u>this is also the direction of propagation of neutrons</u> (now it is, in fact, $\mu_m = \cos\theta_m < 0$); we put therefore:

$$\phi_{i-1/2,m} = 2\phi_{i,m} - \phi_{i+1/2,m}$$

  that, introduced into ($\circ$) allows to obtain the central flux in the form:

$$\phi_{i,m} = \frac{\phi_{i+1/2,m} + \frac{\Delta x}{2|\mu_m|} q_{i,m}}{1 + \frac{\Delta x}{2|\mu_m|}\Sigma_{ti}}$$

  once the central flux is known, it is therefore possible to evaluate the interfacial flux $\phi_{i-1/2,m} = 2\phi_{i,m} - \phi_{i+1/2,m}$ to be used in the calculation of the next subinterval.

- **The order of accuracy obtained by the diamond rule can be analysed considering the particular case of zero emission density and constant total cross section**

$$\mu_m \frac{d\phi(x,\mu_m)}{dx} + \Sigma_t \phi(x,\mu_m) = 0$$

  whose exact solution is

$$\phi(x,\mu_m) = \phi(x',\mu_m)\, e^{-\Sigma_t (x-x')/\mu_m}$$

  In particular, putting $x' = x_{i-1/2}$ and $x = x_{i+1/2}$ it is

$$\phi_{i+1/2,m} = \phi_{i-1/2,m} e^{-h}$$

  where

$$h = \frac{\Sigma_t \Delta x}{\mu_m}$$

Making use of the relations

$$\phi_{i,m} = \frac{\phi_{i-1/2,m} + \frac{\Delta x}{2\mu_m} q_{i,m}}{1 + \frac{\Delta x}{2\mu_m}\Sigma_{ti}} \qquad \textbf{and} \quad \phi_{i+1/2,m} = 2\phi_{i,m} - \phi_{i-1/2,m}$$

it is finally found $\phi_{i+1/2,m} = \phi_{i-1/2,m} \frac{1-h/2}{1+h/2}$

to be considered in view that $e^{-h} = \frac{1-h/2}{1+h/2} + O\left(h^3\right)$

- **Notwithstanding the high accuracy of the method, considering the above relationship it can be noted that when $h > 2$ it is $\phi_{i+1/2} < 0$ even if $\phi_{i-1/2} > 0$**
- **It is a typical problem of this method encountered during calculation advancement that occurs when the relationships**

$$\phi_{i+1/2,m} = 2\phi_{i,m} - \phi_{i-1/2,m} \qquad \mu_m > 0$$
$$\phi_{i-1/2,m} = 2\phi_{i,m} - \phi_{i+1/2,m} \qquad \mu_m < 0$$

  provide negative values of the interface flux
- **The problem can be solved by using a finer spatial discretisation, in order to get $h < 2$; however, this is not always convenient, e.g. in the case of strongly absorbing regions and/or very much inclined direactions**
- **However, it is possible to correct ("fix") the flux making use of one of two simple rules for "fix-up"**
  - **<u>1st RULE</u> ("step method")**
    for $\mu_m > 0$ whenever it is $\phi_{i+1/2,m} = 2\phi_{i,m} - \phi_{i-1/2,m} < 0$, it is assumed $\phi_{i+1/2,m} = \phi_{i,m}$ (instead of the diamond rule), then calculating $\phi_{i,m}$ as a consequence of this choice by (°); similarly in case of $\mu_m < 0$,… (just exchange the role of the two interfaces);
  - **<u>2nd RULE</u> ("set offending flux to zero and recompute")**
    for $\mu_m > 0$ whenever it is $\phi_{i+1/2,m} = 2\phi_{i,m} - \phi_{i-1/2,m} < 0$, it is assumed $\phi_{i+1/2,m} = 0$ (instead of the diamond rule), then calculating $\phi_{i,m}$ as a consequence of this choice by (°);similarly in case of $\mu_m < 0$, … (just exchange the role of the two interfaces).
- **Unfortunately the use of these rules decreases the accuracy of the method from the second order to the first one**

- **Let us just note that the assumption of an isotrpic scattering source is not at all needed for the application of the above describe algorithm; on the contrary, discrete ordinates methods are particularly suitable for dealing with scattering anisotropy**
- **In fact, when the anisotrpy of scattering in the laboratory reference frame is up to the order L, making use of the expansion in Legendre polynomials, we have**

$$\mu \frac{\partial \phi(x,\mu)}{\partial x} + \Sigma_t(x)\phi(x,\mu) = \sum_{l=0}^{L} \frac{2l+1}{2} \Sigma_{sl}(x) P_l(\mu)\phi_l(x) + S(x,\mu)$$

with $$\phi_l(x) = \int_{-1}^{1} P_l(\mu')\phi(x,\mu')d\mu'$$

- **The discrete ordinate form of this equation is therefore given by**

$$\mu_m \frac{\partial \phi(x,\mu_m)}{\partial x} + \Sigma_t(x)\phi(x,\mu_m) = \sum_{l=0}^{L} \frac{2l+1}{2} \Sigma_{sl}(x) P_l(\mu_m)\phi_l(x) + S(x,\mu_m) \qquad (m = 1,\ldots,M)$$

$$\phi_l(x) = \sum_{n=0}^{M} w_n P_l(\mu_n)\phi(x,\mu_n)$$

- **On the basis of this relationship, it is then possible to set up a solution algorithm quite similar to the one just described for the case of the isotropic scattering**

The one-dimensional case in spherical coordinates

Form of the transport equation

- **In the figure reported in the next page, an unfortunate feature of the transport equation in curvilinear coordinates is described; it consists in the fact that the angular coordinate identifying the direction of neutrons changes during the rectilinear motion of neutron**
- **As a consequence of this phenomenon, known as *angular redistribution*, the "streaming" term of the integro-differential equation involves derivatives in the angular coordinate**
- **In order to obtain the streaming term in spherical coordinates, it is necessary to remember that it represents a differentiation along the direction of motion of neutrons**

- **In fact, it is**

$$\vec{\Omega} \cdot grad_{\vec{r}} \phi = \frac{d\phi}{ds}$$

as it can be recognised by putting

$$\vec{r} = \vec{r}(s) = \vec{r}_0 + \vec{\Omega} s \equiv \{x(s), y(s), z(s)\}$$

with

$$x(s) = x_0 + \Omega_x s \qquad y(s) = y_0 + \Omega_y s \qquad z(s) = z_0 + \Omega_z s$$

- **In spherical geometry, the radius and the cosine of the angle between the radius and the direction of motion are taken as independent coordinates:**

$$r \equiv \sqrt{x^2 + y^2 + z^2} \qquad \mu \equiv \cos\theta = \vec{\Omega} \cdot \frac{\vec{r}}{r}$$

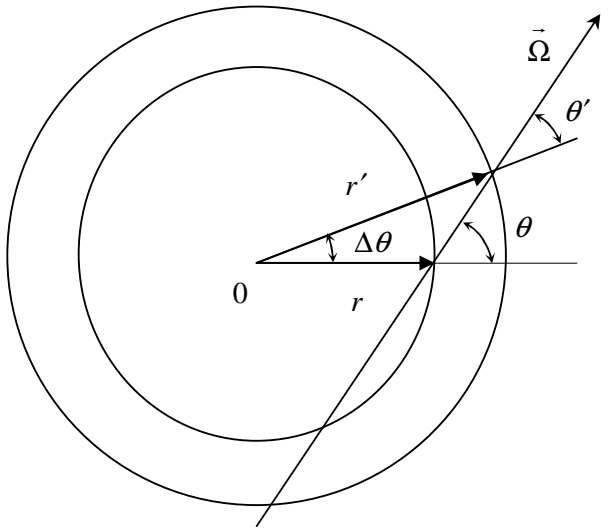
- **Making use of these coordinates, we can therefore write**

$$\vec{\Omega} \cdot grad_{\vec{r}} \phi = \frac{d\phi}{ds} = \frac{\partial \phi}{\partial r}\frac{dr}{ds} + \frac{\partial \phi}{\partial \mu}\frac{d\mu}{ds}$$

translating the problem into the one of expressing the derivatives of r and μ with respect to s

- **Concerning the derivative of r we have**

$$\frac{dr}{ds} = \frac{\partial r}{\partial x}\frac{dx}{ds} + \frac{\partial r}{\partial y}\frac{dy}{ds} + \frac{\partial r}{\partial z}\frac{dz}{ds}$$



Curvilinear coordinates and angular redistribution

- **Since making use of the previous definitions it is**

$$\frac{\partial r}{\partial x}=\frac{x}{r} \qquad \frac{\partial r}{\partial y}=\frac{y}{r} \qquad \frac{\partial r}{\partial z}=\frac{z}{r}$$

and

$$\frac{dx}{ds}=\Omega_x \qquad \frac{dy}{ds}=\Omega_y \qquad \frac{dz}{ds}=\Omega_z$$

it is

$$\frac{dr}{ds}=\frac{x}{r}\Omega_x+\frac{y}{r}\Omega_y+\frac{z}{r}\Omega_z=\vec{\Omega}\cdot\frac{\vec{r}}{r}=\mu$$

- **Similarly, for the derivative with respect to μ we have:**

$$\frac{d\mu}{ds}=\frac{d}{ds}\left(\vec{\Omega}\cdot\frac{\vec{r}}{r}\right)=\frac{\vec{\Omega}}{r}\cdot\left\{\frac{dx}{ds}\vec{i}+\frac{dy}{ds}\vec{j}+\frac{dz}{ds}\vec{k}\right\}-\frac{\vec{\Omega}\cdot\vec{r}}{r^2}\left(\frac{\partial r}{\partial x}\frac{dx}{ds}+\frac{\partial r}{\partial y}\frac{dy}{ds}+\frac{\partial r}{\partial z}\frac{dz}{ds}\right)$$

$$=\frac{\vec{\Omega}}{r}\cdot\left\{\Omega_x\vec{i}+\Omega_y\vec{j}+\Omega_z\vec{k}\right\}-\frac{\vec{\Omega}\cdot\vec{r}}{r^2}\left(\frac{x}{r}\Omega_x+\frac{y}{r}\Omega_y+\frac{z}{r}\Omega_z\right)$$

$$=\frac{1}{r}-\frac{\left(\vec{\Omega}\cdot\vec{r}\right)^2}{r^3}=\frac{1}{r}\left[1-\left(\frac{\vec{\Omega}\cdot\vec{r}}{r}\right)^2\right]=\frac{1-\mu^2}{r}$$

- **The transport equation in spherical coordinates for the monokinetic case and isotropic scattering becomes therefore**

$$\mu\frac{\partial\phi(r,\mu)}{\partial r}+\frac{1-\mu^2}{r}\frac{\partial\phi(r,\mu)}{\partial\mu}+\Sigma_t(r)\phi(r,\mu)=\frac{\Sigma_s(r)}{2}\int_{-1}^{1}\phi(r,\mu')d\mu'+S(r,\mu)$$

- **In view of the spatial and angular discretisation, it is convenient to recast the streaming term into a *conservative form*, i.e., in a form that allows integration over a finite volume of the coordinates with "exact" neutron conservation**

- **In spherical geometry, the control volumes on which we need to integrate are spherical shells; by integrating the streaming term over the general shell with inner and outer radiuses r_1 and r_2 and over all directions, we have:**

$$\iint\vec{\Omega}\cdot grad_{\vec{r}}\phi\,d\Omega\,dV=\int_{r_1}^{r_2}4\pi r^2dr\int_{-1}^{1}2\pi\nabla\cdot\left(\vec{\Omega}\phi\right)d\mu$$

$$=\int_{r_1}^{r_2}4\pi r^2\nabla\cdot\vec{J}(r)\,dr=4\pi r_2^2J(r_2)-4\pi r_1^2J(r_1)$$

- **In order to make this result be obtained easily after multiplication by $4\pi r^2$, the streaming term is rewritten as**

$$\frac{\mu}{r^2}\frac{\partial}{\partial r}\left(r^2\phi\right)+\frac{1}{r}\frac{\partial}{\partial\mu}\left[\left(1-\mu^2\right)\phi\right] \qquad \left\{=\mu\frac{\partial\phi}{\partial r}+\frac{2}{r}\mu\phi-\frac{2}{r}\mu\phi+\frac{1-\mu^2}{r}\frac{\partial\phi}{\partial\mu}=\mu\frac{\partial\phi}{\partial r}+\frac{1-\mu^2}{r}\frac{\partial\phi}{\partial\mu}\right\}$$

It is

$$\int_{r_1}^{r_2} 4\pi r^2 dr \int_{-1}^{1} 2\pi \left\{ \frac{\mu}{r^2}\frac{\partial}{\partial r}\left(r^2\phi\right) + \frac{1}{r}\frac{\partial}{\partial \mu}\left[\left(1-\mu^2\right)\phi\right] \right\} d\mu$$

$$= \int_{r_1}^{r_2} 4\pi \frac{\partial}{\partial r} \underbrace{\left(r^2 \int_{-1}^{1} 2\pi \mu \phi \, d\mu \right)}_{r^2 J} dr + \int_{r_1}^{r_2} 8\pi^2 r dr \underbrace{\int_{-1}^{1} \frac{\partial}{\partial \mu}\left[\left(1-\mu^2\right)\phi\right] d\mu}_{=0}$$

$$= \int_{r_1}^{r_2} 4\pi \frac{\partial}{\partial r}\left(r^2 J\right) dr = 4\pi\left[r_2^2 J(r_2) - r_1^2 J(r_1)\right]$$

- **The *conservative* form of the transport equation in spherical coordinates is therefore:**

$$\frac{\mu}{r^2}\frac{\partial}{\partial r}\left(r^2\phi\right) + \frac{1}{r}\frac{\partial}{\partial \mu}\left[\left(1-\mu^2\right)\phi\right] + \Sigma_t(r)\phi(r,\mu) = \frac{\Sigma_s(r)}{2}\int_{-1}^{1}\phi(r,\mu')d\mu' + S(r,\mu)$$

- **Finally, introducing the emission density**

$$q(r,\mu) = \frac{\Sigma_s(r)}{2}\int_{-1}^{1}\phi(r,\mu')d\mu' + S(r,\mu)$$

we have

$$\frac{\mu}{r^2}\frac{\partial}{\partial r}\left(r^2\phi\right) + \frac{1}{r}\frac{\partial}{\partial \mu}\left[\left(1-\mu^2\right)\phi\right] + \Sigma_t(r)\phi(r,\mu) = q(r,\mu) \qquad (^\circ)$$

Discretised equations

- **In similarity with the Cartesian plane case, also in spherical geometry there is no variation of the angular flux with the angle φ**
- **Therefore, the angular discretisation affects only μ**
- **The spatial discretisation is made in similarity with what already observed for the plane case in Cartesian coordinates**
- **The rectangular discretisation domain shown in the Figure at the bottom of this page is so obtained, where the "diamond", giving the name to the already mentioned rule, is clearly shown**
- **By integrating ($^\circ$) on this domain and in $d\varphi$ over $0 < \varphi < 2\pi$, it is:**

$$\int_0^{2\pi} d\varphi \int_{\mu_{m-1/2}}^{\mu_{m+1/2}} d\mu \int_{r_{i-1/2}}^{r_{i+1/2}} \left\{ \frac{\mu}{r^2}\frac{\partial\left(r^2\phi\right)}{\partial r} + \frac{1}{r}\frac{\partial\left[\left(1-\mu^2\right)\phi\right]}{\partial \mu} + \Sigma_t\phi - q \right\} 4\pi r^2 dr = 0$$

from which we have

$$2\pi\int_{\mu_{m-1/2}}^{\mu_{m+1/2}} \mu\left[4\pi r_{i+1/2}^2\phi\left(r_{i+1/2},\mu\right) - 4\pi r_{i-1/2}^2\phi\left(r_{i-1/2},\mu\right)\right]d\mu$$

$$+ 8\pi^2\int_{r_{i-1/2}}^{r_{i+1/2}}\left[\left(1-\mu_{m+1/2}^2\right)\phi\left(r,\mu_{m+1/2}\right) - \left(1-\mu_{m-1/2}^2\right)\phi\left(r,\mu_{m-1/2}\right)\right] r\, dr$$

$$+ 2\pi\int_{\mu_{m-1/2}}^{\mu_{m+1/2}} d\mu \int_{r_{i-1/2}}^{r_{i+1/2}} \left(\Sigma_t\phi - q\right) 4\pi r^2 dr = 0$$

- **It can be noted that the use of the conservative form of the transport equation allowed a quite easy integration**
- **The three obtained integral terms are then approximated. For the first term, it is**

$$2\pi\int_{\mu_{m-1/2}}^{\mu_{m+1/2}}\mu\left[4\pi r_{i+1/2}^2\phi\left(r_{i+1/2},\mu\right)-4\pi r_{i-1/2}^2\phi\left(r_{i-1/2},\mu\right)\right]d\mu$$
$$\cong 4\pi w_m\mu_m\left(A_{i+1/2}\phi_{i+1/2,m}-A_{i-1/2}\phi_{i-1/2,m}\right)$$

where we assumed

$$w_m=\frac{1}{2}\left(\mu_{m+1/2}-\mu_{m-1/2}\right)\qquad \mu_m=\frac{1}{2}\left(\mu_{m+1/2}+\mu_{m-1/2}\right)\qquad A_{i\pm1/2}=4\pi r_{i\pm1/2}^2$$

and $\phi_{i\pm1/2,m}$ **represents the mean angular flux on the angular element** $\Delta\Omega_m=4\pi w_m$ **holding fot the two surfaces at the radiuses** $r_{i-1/2}$ **or** $r_{i+1/2}$**:**

$$\phi_{i\pm1/2,m}=\frac{1}{4\pi w_m}\int_0^{2\pi}d\varphi\int_{\mu_{m-1/2}}^{\mu_{m+1/2}}\phi\left(r_{i\pm1/2},\mu\right)d\mu$$

- **The second term is now approximated as:**

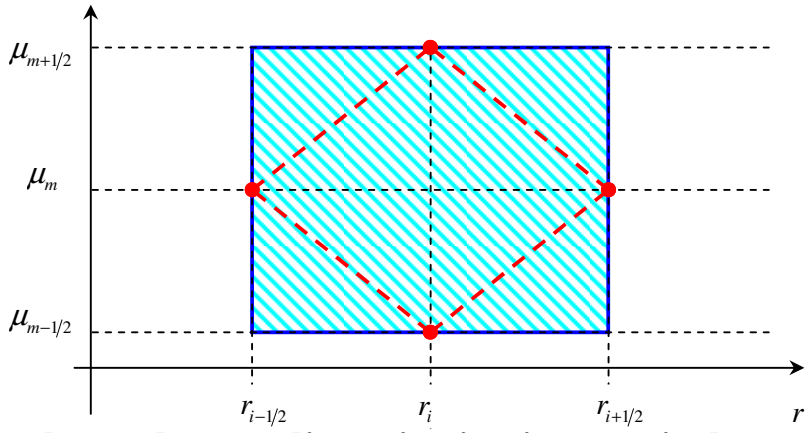
$$8\pi^2\int_{r_{i-1/2}}^{r_{i+1/2}}\left[\left(1-\mu_{m+1/2}^2\right)\phi\left(r,\mu_{m+1/2}\right)-\left(1-\mu_{m-1/2}^2\right)\phi\left(r,\mu_{m-1/2}\right)\right]r\,dr$$
$$\approx 4\pi\left(a_{m+1/2}\phi_{i,m+1/2}-a_{m-1/2}\phi_{i,m-1/2}\right)$$

where $\phi_{i,m\pm1/2}$ **is the mean flux over the volume** V_i **(spherical shell between** $r_{i-1/2}$ **and** $r_{i+1/2}$**), corresponding to** $\mu_{i-1/2}$ **and** $\mu_{i+1/2}$

$$\phi_{i,m\pm1/2}=\frac{1}{V_i}\int_{r_{i-1/2}}^{r_{i+1/2}}\phi\left(r,\mu_{m\pm1/2}\right)4\pi r^2dr\qquad V_i=\frac{4}{3}\pi\left(r_{i+1/2}^3-r_{i-1/2}^3\right)$$

while the constants $a_{m-1/2}$ **e** $a_{m+1/2}$ **represent the effect of anfular redistribution, whose value will be described later**

- **The third integral is finally approximated as :**



Angular and space discretisation in spherical geometry

$$2\pi\int_{\mu_{m-1/2}}^{\mu_{m+1/2}} d\mu \int_{r_{i-1/2}}^{r_{i+1/2}} (\Sigma_t \phi - q) 4\pi r^2 dr \cong 4\pi w_m V_i (\Sigma_{ti}\phi_{im} - q_{im})$$

where ϕ_{im} and q_{im} represent mean values of the angular flux and of the emission density over $\Delta\Omega_m$ and V_i in the spherical shell

$$\phi_{im} = \frac{1}{4\pi w_m V_i} \int_0^{2\pi} d\varphi \int_{\mu_{m-1/2}}^{\mu_{m+1/2}} d\mu \int_{r_{i-1/2}}^{r_{i+1/2}} \phi(r,\mu) 4\pi r^2 dr$$

$$q_{im} = \frac{1}{4\pi w_m V_i} \int_0^{2\pi} d\varphi \int_{\mu_{m-1/2}}^{\mu_{m+1/2}} d\mu \int_{r_{i-1/2}}^{r_{i+1/2}} q(r,\mu) 4\pi r^2 dr$$

- **Adopting the above approximations, the discretised neutron transport equation in 1D spherical geometry takes the form**

$$\mu_m \left(A_{i+1/2}\phi_{i+1/2,m} - A_{i-1/2}\phi_{i-1/2,m}\right) + \frac{1}{w_m}\left(a_{m+1/2}\phi_{i,m+1/2} - a_{m-1/2}\phi_{i,m-1/2}\right) + V_i(\Sigma_{ti}\phi_{im} - q_{im}) = 0$$

- **In order to obtain the angular discretisation coefficients, one proceeds as follows:**
  - **it is noted that for <u>uniform and istropic flux</u> it is**

$$\phi_{i+1/2,m} = \phi_{i-1/2,m} = \phi_{i,m}$$

requiring that streaming term is zero, so that

$$\Sigma_{ti}\phi_{im} = q_{im} \;;$$

then, it must be requested that

$$\mu_m w_m \left(A_{i+1/2} - A_{i-1/2}\right) = a_{m-1/2} - a_{m+1/2} \qquad (*)$$

  - **however, in the general case the term depending on the angular derivative must become zero when integrated over all the directions; therefore, since it must be**

$$\sum_{m=1}^{M}\left(a_{m+1/2}\phi_{i,m+1/2} - a_{m-1/2}\phi_{i,m-1/2}\right) = a_{M+1/2}\phi_{i,M+1/2} - a_{1/2}\phi_{i,1/2} = 0 \;;$$

since $\phi_{i,1/2}$ and $\phi_{i,M+1/2}$ are arbitrary, it must be also

$$a_{1/2} = a_{M+1/2} = 0$$

- **So, assuming $a_{1/2} = 0$ the equation (*) allows to calculate all the coefficients a_m by recurrence.**

Solution algorithm

- **The directions are chosen so that**

$$-1 = \mu_{1/2} < \mu_1 < \mu_{3/2} < \ldots < \mu_{M/2} < \mu_{M/2+1/2} = 0 < \mu_{M/2+1} < \ldots < \mu_{M-1/2} < \mu_M < \mu_{M+1/2} = 1$$

where $\mu_{1/2} = -1$ is said *starting direction*, since it is the direction for starting the calcualtion

- **In fact, for a free surface (interface with the void), the boundary conditions are**

$$\phi_{I+1/2,m} = 0 \qquad (m = 1,2,...,M/2)$$

assuming zero the angular flux related to the directions with negative μ (i.e., "inward" directions)

- **In the center of the sphere, we must assume that the angular flux satisfies symmetry conditions generally expressed by imposing**

$$\phi_{1/2,M+1-m} = \phi_{1/2,m} \qquad (m = 1,2,...,M/2)$$

- **Then, the calculation starts with the direction $\mu_{1/2} = -1$: since the streaming term does not contain the angular dispersion term, being $1-\mu^2 = 0$, we then have:**

$$-\frac{\phi_{i+1/2,1/2} - \phi_{i-1/2,1/2}}{r_{i+1/2} - r_{i-1/2}} + \Sigma_{ti}\phi_{i,1/2} = q_{i,1/2}$$

in which usally the *diamond rule* is adopted in the form

$$\phi_{i-1/2,1/2} = 2\phi_{i,1/2} - \phi_{i+1/2,1/2}$$

in order to advance the calculation in the direction of decreasing r starting from the boundary condition of zero inward flux on the external surface

- **One then proceeds with all the directions $1 \le m \le M/2$ (for $\mu < 0$) <u>adopting the diamond rule in the twofold spatial and angular form</u>:**

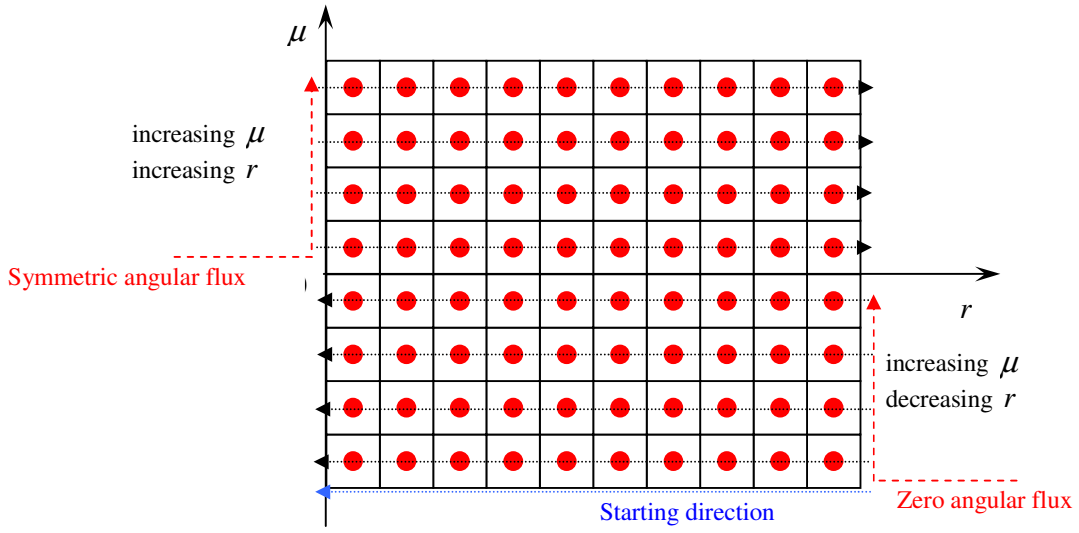
$$\phi_{im} \cong \frac{\phi_{i-1/2,m} + \phi_{i+1/2,m}}{2} \cong \frac{\phi_{i,m-1/2} + \phi_{i,m+1/2}}{2}$$

Since the process is performed for decreasing r and increasing μ, the diamond rule is adopted in the two forms

$$\phi_{i-1/2,m} = 2\phi_{im} - \phi_{i+1/2,m} \qquad \phi_{i,m+1/2} = 2\phi_{im} - \phi_{i,m-1/2}$$

that, introduced into the discretised transport equation, allow to eliminate the unkown values of the angular flux

$$\phi_{im} = \frac{-\mu_m\left(A_{i-1/2} + A_{i+1/2}\right)\phi_{i+1/2,m} + \frac{1}{w_m}\left(a_{m-1/2} + a_{m+1/2}\right)\phi_{i,m-1/2} + V_i q_{im}}{-\mu_m\left(A_{i-1/2} + A_{i+1/2}\right) + \frac{1}{w_m}\left(a_{m-1/2} + a_{m+1/2}\right) + V_i\Sigma_{ti}}$$



Sample advancement scheme for an S_8 method with 10 radial nodes

where use is made of the definition of the angular differentiation coefficients

- **Once the values of the angular flux for all the directions with $\mu < 0$ are computed, the following symmetry condition is used**

$$\phi_{1/2,M+1-m} = \phi_{1/2,m} \qquad (m = 1,2,\ldots, M/2)$$

in order to assigne the flux in the centre of the sphere for the directions with $\mu > 0$. The calculations then proceeds for increasing r and μ, making use of the diamond rule in the two forms

$$\phi_{i+1/2,m} = 2\phi_{im} - \phi_{i-1/2,m} \qquad \phi_{i,m+1/2} = 2\phi_{im} - \phi_{i,m-1/2}$$

- **Even in the spherical case, it is possible to encounter problems related to the negative fluxes requiring the use of "fix-up" rules**

- **In some cases a non completely correct behaviour of the flux in the centre of the sphere has be noted, to be attributed to a non-uniform distribution of truncation erroras a function of r**

- **The condition**

$$\phi_{1/2,M+1-m} = \phi_{1/2,m} \qquad (m = 1,2,\ldots, M/2)$$

has been also considered criticisable, preferring sometimes the relationship

$$\left.\frac{\partial \phi}{\partial \mu}\right|_{r \to 0} = 0$$

discretised imposing that the angular flux at the centre of the sphere is equal in all the directions

The multidimensional case in Cartesian coordinates

Discretised equations

- **In the multidimensional cases it is convenient to write the integro-differential equation in the form**

$$div\left[\vec{\Omega}\phi\left(\vec{r},\vec{\Omega}\right)\right] + \Sigma_t\left(\vec{r}\right)\phi\left(\vec{r},\vec{\Omega}\right) - q\left(\vec{r},\vec{\Omega}\right) = 0$$

where again the following relationship has been used

$$\vec{\Omega}\cdot grad_{\vec{r}}\phi\left(\vec{r},\vec{\Omega}\right) = div\left[\vec{\Omega}\phi\left(\vec{r},\vec{\Omega}\right)\right]$$

- **Considering a volume around the location $\vec{r}_{ijk} = \{x_i, y_j, z_k\}$, it is**

$$V_{ijk} = \Delta x_i \Delta y_j \Delta z_k \qquad A_{i-1/2,jk} = A_{i+1/2,jk} = \Delta y_j \Delta z_k = \frac{V_{ijk}}{\Delta x_i}$$

$$A_{i,j-1/2,k} = A_{i,j+1/2,k} = \Delta x_i \Delta z_k = \frac{V_{ijk}}{\Delta y_j} \qquad A_{ij,k-1/2} = A_{ij,k+1/2} = \Delta x_i \Delta y_j = \frac{V_{ijk}}{\Delta z_k}$$

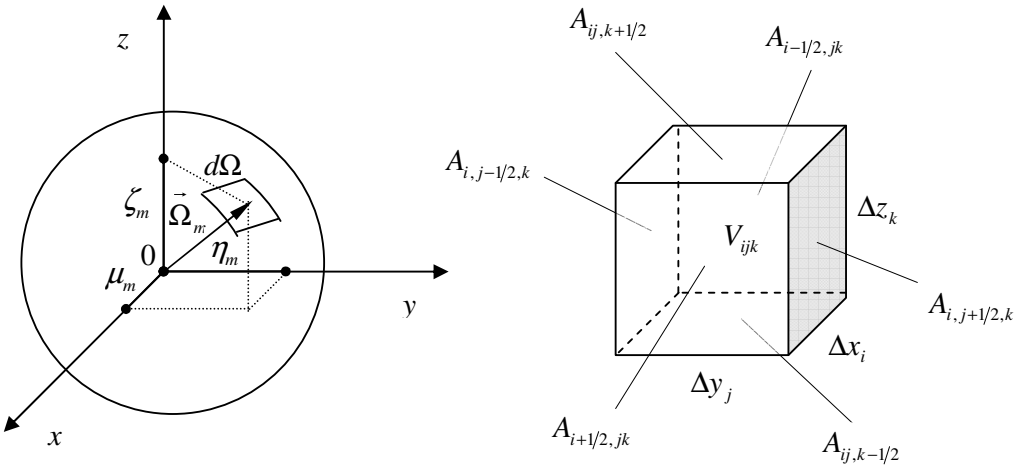
$$(i = 1,\ldots,I);\ (j = 1,\ldots,J);\ (k = 1,\ldots,K)$$

- **To the general direction, $\vec{\Omega}_m$, the solid angle $\Delta\Omega_m = 4\pi w_m$ is then assigned. We then define the average values of the flux over the solid angle and the volume and also on the volume faces**

$$\phi_{ijk,m} = \frac{1}{4\pi w_m V_{ijk}} \int_{4\pi w_m} d\Omega \int_{V_{ijk}} \phi\, dV \qquad \phi_{i\pm1/2,jk,m} = \frac{1}{4\pi w_m A_{i\pm1/2,jk}} \int_{4\pi w_m} d\Omega \int_{A_{i\pm1/2,jk}} \phi dA$$

$$\phi_{i,j\pm1/2,k,m} = \frac{1}{4\pi w_m A_{i,j\pm1/2,k}} \int_{4\pi w_m} d\Omega \int_{A_{i,j\pm1/2,k}} \phi dA \qquad \phi_{ij,k\pm1/2,m} = \frac{1}{4\pi w_m A_{ij,k\pm1/2}} \int_{4\pi w_m} d\Omega \int_{A_{ij,k\pm1/2}} \phi dA$$

- **By integrating the transport equation over the solid angle $\Delta\Omega_m = 4\pi w_m$ and over V_{ijk} we have:**



Reference frame and elementary volume

$$\int_{4\pi w_m} d\Omega \int_{V_{ijk}} \left\{ div\left[\vec{\Omega}\phi\left(\vec{r},\vec{\Omega}\right)\right] + \Sigma_t\left(\vec{r}\right)\phi\left(\vec{r},\vec{\Omega}\right) - q\left(\vec{r},\vec{\Omega}\right) \right\} dV = 0$$

- **The first term in this equation can be approximated as follows:**

$$\int_{4\pi w_m} d\Omega \int_{V_{ijk}} div\left[\vec{\Omega}\phi\left(\vec{r},\vec{\Omega}\right)\right]\, dV \cong \int_{4\pi w_m} d\Omega \int_{V_{ijk}} div\left[\vec{\Omega}_m\phi\left(\vec{r},\vec{\Omega}_m\right)\right]\, dV = \int_{4\pi w_m} d\Omega \int_{\partial V_{ijk}} \left(\vec{\Omega}_m \cdot \vec{u}_e\right)\phi\left(\vec{r},\vec{\Omega}_m\right) dA$$

- **Then, putting**

$$\vec{\Omega} \equiv \{\mu_m, \eta_m, \zeta_m\}$$

we have

$$\int_{4\pi w_m} d\Omega \int_{\partial V_{ijk}} \left(\vec{\Omega}_m \cdot \vec{u}_e\right)\phi\left(\vec{r},\vec{\Omega}_m\right) dA \cong 4\pi w_m \left\{ \mu_m \left(A_{i+1/2,\,jk}\phi_{i+1/2\,jk,m} - A_{i-1/2,\,jk}\phi_{i-1/2\,jk,m} \right) \right.$$

$$\left. + \eta_m \left(A_{i.\,j+1/2,k}\phi_{ij+1/2k,m} - A_{i,\,j-1/2,k}\phi_{ij-1/2k,m} \right) + \zeta_m \left(A_{ij,k+1/2}\phi_{ijk+1/2,m} - A_{ij,k-1/2}\phi_{ijk-1/2,m} \right) \right\}$$

- **Considering the previous definitions, it is:**

$$\int_{4\pi w_m} d\Omega \int_{\partial V_{ijk}} \left(\vec{\Omega}_m \cdot \vec{u}_e\right)\phi\left(\vec{r},\vec{\Omega}_m\right) dA \cong 4\pi w_m V_{ijk} \left\{ \mu_m \frac{\phi_{i+1/2\,jk,m} - \phi_{i-1/2\,jk,m}}{\Delta x_i} \right.$$

$$\left. + \eta_m \frac{\phi_{i,\,j+1/2k,m} - \phi_{i,\,j-1/2k,m}}{\Delta y_j} + \zeta_m \frac{\phi_{ijk+1/2,m} - \phi_{ijk-1/2,m}}{\Delta z_k} \right\}$$

- **It is also put**

$$\int_{4\pi w_m} d\Omega \int_{V_{ijk}} \left\{ \Sigma_t\left(\vec{r}\right)\phi\left(\vec{r},\vec{\Omega}\right) - q\left(\vec{r},\vec{\Omega}\right) \right\} dV \cong 4\pi w_m V_{ijk} \left(\Sigma_{t,ijk}\phi_{ijk,m} - q_{ijk,m} \right)$$

where it is

$$q_{ijk,m} = \frac{1}{4\pi w_m V_{ijk}} \int_{4\pi w_m} d\Omega \int_{V_{ijk}} q\left(\vec{r},\vec{\Omega}\right) dV$$

- **We finally have**

$$\mu_m \frac{\phi_{i+1/2\,jk,m} - \phi_{i-1/2\,jk,m}}{\Delta x_i} + \eta_m \frac{\phi_{i,\,j+1/2k,m} - \phi_{i,\,j-1/2k,m}}{\Delta y_j} + \zeta_m \frac{\phi_{ijk+1/2,m} - \phi_{ijk-1/2,m}}{\Delta z_k}$$

$$+ \Sigma_{t,ijk}\phi_{ijk,m} = q_{ijk,m} \qquad (i = 1,\ldots,I);(j = 1,\ldots,J);(k = 1,\ldots,K);(m = 1,\ldots,N_m)$$

representing the discrete ordinate form of the transport equation for the multi-dimensional case

Solution algorithm and choice of directions

- **Also in this case the solution algorithm takes into account the direction of motion of neutrons and then of the sign of direction cosines μ_m, η_m and ζ_m of $\vec{\Omega}_m$**

- **In particular, in anyone of the three directions, the diamond rule is adopted accordint ot the following rationale:**

$$\begin{cases} \mu_m > 0 & \phi_{i+1/2,jk,m} = 2\phi_{ijk,m} - \phi_{i-1/2,jk,m} \\ \mu_m < 0 & \phi_{i-1/2,jk,m} = 2\phi_{ijk,m} - \phi_{i+1/2,jk,m} \end{cases}$$

$$\begin{cases} \eta_m > 0 & \phi_{i,j+1/2,k,m} = 2\phi_{ijk,m} - \phi_{i,j-1/2,k,m} \\ \eta_m < 0 & \phi_{i,j-1/2,k,m} = 2\phi_{ijk,m} - \phi_{i,j+1/2,k,m} \end{cases}$$

$$\begin{cases} \zeta_m > 0 & \phi_{ij,k+1/2,m} = 2\phi_{ijk,m} - \phi_{ij,k-1/2,m} \\ \zeta_m < 0 & \phi_{ij,k-1/2,m} = 2\phi_{ijk,m} - \phi_{ij,k+1/2,m} \end{cases}$$

- **Then, substituting the appropriate form of the diamond rule in the relation**

$$\mu_m \frac{\phi_{i+1/2\,jk,m} - \phi_{i-1/2\,jk,m}}{\Delta x_i} + \eta_m \frac{\phi_{i,j+1/2k,m} - \phi_{i,j-1/2k,m}}{\Delta y_j} + \zeta_m \frac{\phi_{ijk+1/2,m} - \phi_{ijk-1/2,m}}{\Delta z_k}$$
$$+ \Sigma_{t,ijk}\phi_{ijk,m} = q_{ijk,m} \qquad (i = 1,\ldots,I);\ (j = 1,\ldots,J);\ (k = 1,\ldots,K);\ (m = 1,\ldots,N_m)$$

the equations are solved proceeding for increasing or deacreasing coordinates, according to the sign of the respective direction cosine

- **For instance, let us consider to particular cases:**
  - $\mu_m > 0, \eta_m > 0, \zeta_m > 0$

$$\phi_{ijk,m} = \frac{\frac{2\mu_m}{\Delta x_i}\phi_{i-1/2\,jk,m} + \frac{2\eta_m}{\Delta y_j}\phi_{ij-1/2k,m} + \frac{2\zeta_m}{\Delta z_k}\phi_{ijk-1/2,m} + q_{ijk,m}}{\frac{2\mu_m}{\Delta x_i} + \frac{2\eta_m}{\Delta y_j} + \frac{2\zeta_m}{\Delta z_k} + \Sigma_{t,ijk}}$$

  - $\mu_m < 0, \eta_m > 0, \zeta_m < 0$

$$\phi_{ijk,m} = \frac{-\frac{2\mu_m}{\Delta x_i}\phi_{i+1/2\,jk,m} + \frac{2\eta_m}{\Delta y_j}\phi_{ij-1/2k,m} - \frac{2\zeta_m}{\Delta z_k}\phi_{ijk+1/2,m} + q_{ijk,m}}{-\frac{2\mu_m}{\Delta x_i} + \frac{2\eta_m}{\Delta y_j} - \frac{2\zeta_m}{\Delta z_k} + \Sigma_{t,ijk}}$$

<u>Selection of the directions</u>

- **The choice of the directions can be made with a relatively large freedom, though it is generally required that some fundamental criteria are respected**
- **A first criterion consists in imposing that once a given direction $\vec{\Omega}_m = \{\mu_m, \eta_m, \zeta_m\}$ has been chosen, such that it must be**

$$\mu_m^2 + \eta_m^2 + \zeta_m^2 = 1$$

also the directions $\{-\mu_m, \eta_m, \zeta_m\}$, $\{\mu_m, -\eta_m, \zeta_m\}$ and $\{\mu_m, \eta_m, -\zeta_m\}$ are consiered admissible:

this allows imposing in a simple and direct way reflective conditions orthogonal to the there axes of the reference frame

- **Whenever such choice is made, *it is possible to consider only the directions included in a single octant*, then translating the results to the others**
- **A particularly interesting choice is the one consisting in imposing that the adimissible directions are invariant to 90° rotation around any axis of the reference frame**

- **These "*level symmetric quadratures*" are characterisedby the fact of being selected making use of asingle degree of freedom. In fact, it is assumed that the direction cosines are all chosen by a single set, defined as**

$$-1 < -t_{M/2} <..< -t_1 < 0 < t_1 <..< t_{M/2} < 1$$

- **Now, let us assume that we select three cosines such that**

$$\mu_m = t_i \qquad \eta_m = t_j \qquad \zeta_m = t_k$$

In this case, it must be obviously

$$t_i^2 + t_j^2 + t_k^2 = 1 \qquad \textbf{(a)}$$

So, making the further choice $\mu_m = t_i$ and $\eta_m = t_{j+1}$, in order to satisfy the normalisation relationships we require that $\zeta_m = t_{k-1}$:

$$t_i^2 + t_{j+1}^2 + t_{k-1}^2 = 1 \qquad \textbf{(b)}$$

By subtracting side by side *(b)* to *(a)* we get

$$t_{j+1}^2 - t_j^2 = t_k^2 - t_{k-1}^2$$

Since j and k are arbitrary, we have:

$$t_i^2 = t_{i-1}^2 + C \qquad \Rightarrow \qquad t_i^2 = t_1^2 + (i-1)C$$

Finally, since it must be

$$t_1^2 + t_1^2 + t_{M/2}^2 = 1$$

we finally get

$$C = \frac{2\left(1 - 3t_1^2\right)}{M-2}$$

- **Since such quadratures involve for each axis $M/2$ positive and $M/2$ negative values for each direction cosine, they are considered "S_M quadratures" (actually the general name is "S_N")**
- **The total number of directions per each octant is $M(M+2)/8$ and the overall number on the unity radius sphere is $M(M+2)$**

- **For the weighting coefficients, a normalization on each octant is generally adopted**

$$\sum_{m=1}^{M(M+2)/8} w_m^I = 1$$

where w_m^I = is the weighting coefficient for a general direction related to the first octant

- **As a consequence the scalar flux in $\vec{r}_{ijk}$ can be obtained by the angular flux by the relationship**

$$\phi_{ijk} = \frac{1}{8} \sum_{m=1}^{M(M+2)} w_m \phi_{ijk,m}$$

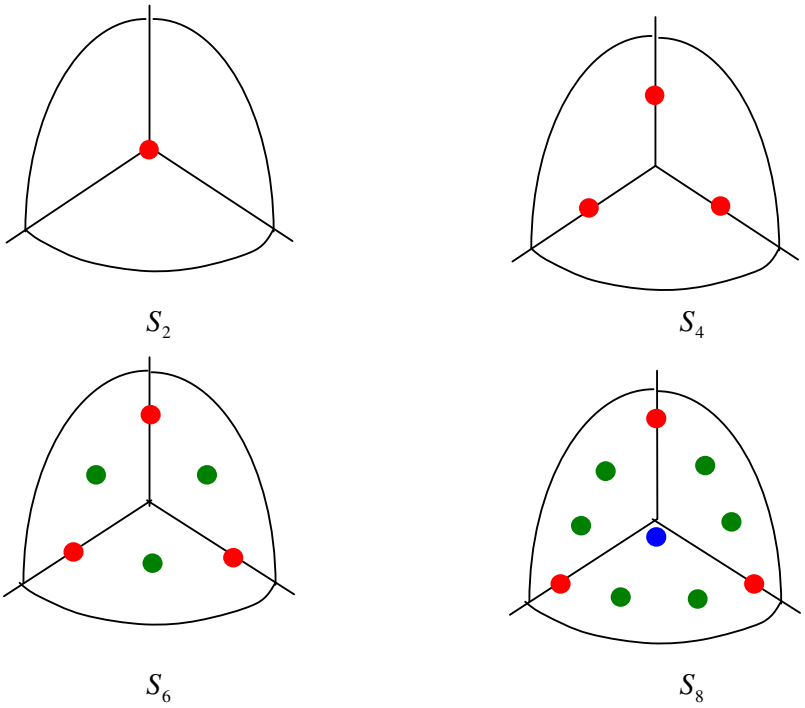
- **In each octant we also assume that the directions obtained by permutation of direction cosines have the same weight**
- **However, even considering all these limitations, it is possible to envisage different choices for the weighting coefficients**
- **For instance, it is possible to request that the maximum possible degree of Legendre polynomials in the three directions be integrated exactly; this leads to the so-called *LQ_n quadratures***

Livello	m	μ_m	w_m
S_2	1	$1/\sqrt{3}$	1
S_4	1	0.3500212	0.3333333
	2	0.8688903	
S_6	1	0.2666355	0.1761263
	2	0.6815076	0.1572071
	3	0.9261808	
S_8	1	0.2182179	0.1209877
	2	0.5773503	0.0907407
	3	0.7867958	0.0925926
	4	0.9511897	

LQ_n parameters for S_N quadratures

- **Otherwise, it can be preferred to assign a fraction of the area of the sphere to any direction, to be used as its weight**

- **The above considerations, related to the 3D case, can be easily applied also to 2D systems; in such a case an S_M method will involve a total number of directions equal to $(M+2)M/2$ in the four quadrants**



Qualitiative indication of the directions for S_N

in the octant with positive direction cosines

Acceleration methods for discrete ordinates

General considerations

- **In the above treatment it was assumed that the emission density, q, was assigned**
- **For purely absorption problems this corresponds to the actual situation, but in most cases having a practical interest the scattering introduces a variability of emission density as a function of flux**
- **As already mentioned, the problem is solved by iterating on the scattering source, starting with an initial guess and adopting an appropriate convergence criterion**
- **As already mentioned, the problem is solved by iterating on the scattering source, starting with an initial guess of angular flux and adopting an appropriate convergence criterion**

- **However, convergence may not be fast in cases of optically thick regions and when the scattering within a given energy group is considerable**
- **In such cases, an acceleration procedure is necessary**

"Coarse mesh-rebalance"

- **This technique is based on the fact that *a distribution of angular flux obtained after reaching convergence must satisfy the neutron balance***
- **On the contrary, this is generally not true for the angular flux obtained at some iteration, which is based on the scattering source guessed at the previous iteration**
- **The basic idea of the method is therefore to modify the angular flux distribution by multiplying it by variable factors to be determined in "macro-regions" (coarse meshes) just imposing the neutron balance**
- **As we already discussed, the neutron continuity equation can be obtained from the integro-differential equation integrated over the whole solid angle:**

$$\int_{4\pi} \vec{\Omega} \cdot grad_{\vec{r}} \phi(\vec{r},\vec{\Omega}) d\Omega + \int_{4\pi} \Sigma_t(\vec{r}) \phi(\vec{r},\vec{\Omega}) d\Omega = \int_{4\pi} d\Omega \int_{4\pi} \Sigma_s(\vec{r},\vec{\Omega}\cdot\vec{\Omega}') \phi(\vec{r},\vec{\Omega}') d\Omega' + \int_{4\pi} S(\vec{r},\vec{\Omega}) d\Omega$$

and then

$$div \int_{4\pi} \vec{\Omega} \phi(\vec{r},\vec{\Omega}) d\Omega + \Sigma_t(\vec{r}) \int_{4\pi} \phi(\vec{r},\vec{\Omega}) d\Omega = \int_{4\pi} \phi(\vec{r},\vec{\Omega}') d\Omega' \int_{4\pi} \Sigma_s(\vec{r},\vec{\Omega}\cdot\vec{\Omega}') d\Omega + \int_{4\pi} S(\vec{r},\vec{\Omega}) d\Omega$$

and, again

$$div \vec{J}(\vec{r}) + \Sigma_t(\vec{r}) \phi(\vec{r}) = \Sigma_s(\vec{r}) \phi(\vec{r}) + S(\vec{r})$$

or

$$div \vec{J}(\vec{r}) + \Sigma_r(\vec{r}) \phi(\vec{r}) = S(\vec{r})$$

where we have put $\Sigma_r(\vec{r}) = \Sigma_t(\vec{r}) - \Sigma_s(\vec{r})$

- **The integration domain, already subdivided into many relatively small nodes for the solution of the transport equation by the S_N method, is now subdivided into N_m larger regions ("coarse meshes")**
- **We than impose that the neutron balance is satisfied in every region V_m**

$$\int_{V_m} div\vec{J}(\vec{r})dV + \int_{V_m} \Sigma_r(\vec{r})\phi(\vec{r})dV = \int_{V_m} S(\vec{r})dV$$

obtaining

$$\sum_{m'} \int_{\Gamma_{mm'}} \vec{J}(\vec{r})\cdot\vec{u}_e d\Gamma + \int_{V_m} \Sigma_r(\vec{r})\phi(\vec{r})dV = \int_{V_m} S(\vec{r})dV$$

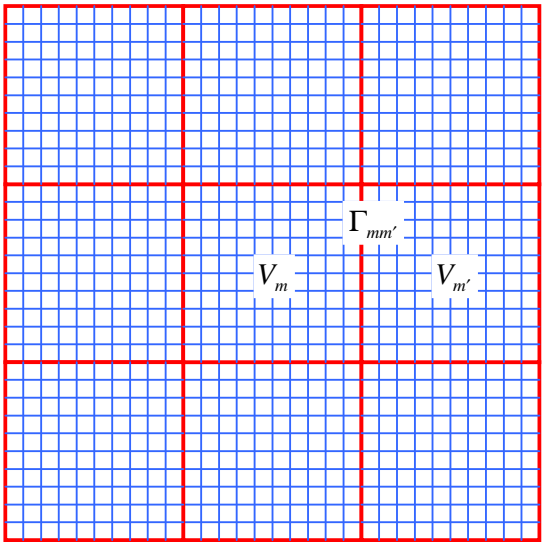
- **For reasons that will be clear in a while, it is convenient to subdivide the current at each interface between adjoining regions into the "inward" and the "outward" contributions. It is therefore:**

$$\sum_{m'} \int_{\Gamma_{mm'}} J_+(\vec{r})d\Gamma - \sum_{m'} \int_{\Gamma_{mm'}} J_-(\vec{r})d\Gamma + \int_{V_m} \Sigma_r(\vec{r})\phi(\vec{r})dV = \int_{V_m} S(\vec{r})dV$$

- **In this equation, we now assume that the scalar flux and the currents are numerically obtained by integrating the angular flux obtained at the l-th iteration by the S_N method**
- **Identifying this angular flux distribution by $\tilde{\phi}^l(\vec{r},\vec{\Omega})$, we assume that it can be multiplied by a coefficient (presently unknown) that is different for each region. The purpose of this action is to impose the fulfillment of the neuotrn balance:**

$$\begin{aligned}
\phi^{l+1}(\vec{r},\vec{\Omega}) &= f_m\tilde{\phi}^l(\vec{r},\vec{\Omega}) && \vec{r}\in V_m \\
\phi^{l+1}(\vec{r},\vec{\Omega}) &= f_m\tilde{\phi}^l(\vec{r},\vec{\Omega}) && \vec{r}\in\Gamma_{mm'},\vec{\Omega}\cdot\vec{u}_e>0 \\
\phi^{l+1}(\vec{r},\vec{\Omega}) &= f_{m'}\tilde{\phi}^l(\vec{r},\vec{\Omega}) && \vec{r}\in\Gamma_{mm'},\vec{\Omega}\cdot\vec{u}_e<0
\end{aligned}$$

- **We can therefore put:**



Macro-regions for the "coarse-mesh rebalance"

$$\phi(\vec{r}) = \int_{4\pi} \phi^{l+1}(\vec{r},\vec{\Omega})d\Omega = f_m \int_{4\pi} \tilde{\phi}^{l}(\vec{r},\vec{\Omega})d\Omega = f_m \tilde{\phi}^{l}(\vec{r})$$

$$J_-(\vec{r}) = \int_{\vec{\Omega}\cdot\vec{u}_e<0} \phi^{l+1}(\vec{r},\vec{\Omega})\left|\vec{\Omega}\cdot\vec{u}_e\right| d\Omega = f_{m'} \int_{\vec{\Omega}\cdot\vec{u}_e<0} \tilde{\phi}^{l}(\vec{r},\vec{\Omega})\left|\vec{\Omega}\cdot\vec{u}_e\right| d\Omega = f_{m'} \tilde{J}_-^{l}(\vec{r}) \qquad \vec{r}\in\Gamma_{mm'}$$

$$J_+(\vec{r}) = \int_{\vec{\Omega}\cdot\vec{u}_e>0} \phi^{l+1}(\vec{r},\vec{\Omega})\vec{\Omega}\cdot\vec{u}_e\, d\Omega = f_m \int_{\vec{\Omega}\cdot\vec{u}_e>0} \tilde{\phi}^{l}(\vec{r},\vec{\Omega})\vec{\Omega}\cdot\vec{u}_e\, d\Omega = f_m \tilde{J}_+^{l}(\vec{r}) \qquad \vec{r}\in\Gamma_{mm'}$$

- **By substituting the above formulas in the neutron balance, we have:**

$$\left(\sum_{m'}\int_{\Gamma_{mm'}} \tilde{J}_+^{l}(\vec{r})d\Gamma + \int_{V_m}\Sigma_r(\vec{r})\tilde{\phi}^{l}(\vec{r})dV\right) f_m - \sum_{m'}\left(\int_{\Gamma_{mm'}} \tilde{J}_-^{l}(\vec{r})d\Gamma\right) f_{m'} = \int_{V_m} S(\vec{r})dV$$

and, putting

$$a_{mm} = \sum_{m'}\int_{\Gamma_{mm'}} \tilde{J}_+^{l}(\vec{r})d\Gamma + \int_{V_m}\Sigma_r(\vec{r})\tilde{\phi}^{l}(\vec{r})dV$$

$$a_{mm'} = \int_{\Gamma_{mm'}} \tilde{J}_-^{l}(\vec{r})d\Gamma \qquad\qquad b_m = \int_{V_m} S(\vec{r})dV$$

we have

$$a_{mm} f_m - \sum_{m'\neq m} a_{mm'} f_{m'} = b_m \qquad\qquad (m = 1,2,...,N_m)$$

representing a linear system with sparse matrix in the unknowns f_m

- **The solution of this system allows therefore to obtain the new approximation of the angular flux to be used as a guess of the next iteration cycle on the scattering source**

Diffusion Synthetic Acceleration (DSA)

- **This technique makes use of a low-order approximation of the transport operator in order to improve convergence**
- **For the sake of simplicity, we will restrict the treatment to the case of isotropic scattering and independent source; the neutron transport equation is**

$$\vec{\Omega}\cdot grad_{\vec{r}}\,\phi(\vec{r},\vec{\Omega}) + \Sigma_t(\vec{r})\phi(\vec{r},\vec{\Omega}) - \frac{\Sigma_s(\vec{r})}{4\pi}\int_{4\pi}\phi(\vec{r},\vec{\Omega}')d\Omega' = \frac{S_0(\vec{r})}{4\pi}$$

- **Using an operator notation, we put**

$$H_0\cdot = \vec{\Omega}\cdot grad_{\vec{r}}\cdot + \Sigma_t(\vec{r})\cdot \qquad H_1\cdot = \frac{\Sigma_s(\vec{r})}{4\pi}\int_{4\pi}\cdot\, d\Omega' \qquad H\cdot = H_0\cdot - H_1\cdot$$

obtaining

$$H\phi(\vec{r},\vec{\Omega}) = H_0\phi(\vec{r},\vec{\Omega}) - H_1\phi(\vec{r},\vec{\Omega}) = \frac{S_0(\vec{r})}{4\pi}$$

- **By integrating both sides of the above equation on all the directions, it is**

$$\int_{4\pi} H\phi\left(\vec{r},\vec{\Omega}\right)d\Omega = S_0\left(\vec{r}\right)$$

- **We now introduce the low-order transport operator as the neutron diffusion operator**

$$H_a \cdot = -divD\left(\vec{r}\right)grad_{\vec{r}} \cdot + \Sigma_r\left(\vec{r}\right)\cdot$$

- **The transport operator can be thus written as the summation of the lower order operator plus the difference operator**

$$\int_{4\pi} \left[H_a + \left(H - H_a\right)\right]\phi\left(\vec{r},\vec{\Omega}\right)d\Omega = S_0\left(\vec{r}\right)$$

- **Since the diffusion operator works directly on the scalar flux, the following notation can be adopted**

$$\int_{4\pi} H_a\phi\left(\vec{r},\vec{\Omega}\right)d\Omega = H_a\phi\left(\vec{r}\right)$$

thus obtaining

$$H_a\phi\left(\vec{r}\right) = S_0\left(\vec{r}\right) - \int_{4\pi} \left(H - H_a\right)\phi\left(\vec{r},\vec{\Omega}\right)d\Omega$$

- **This suggests to use the iterative scheme**

$$H_a\phi^{l+1}\left(\vec{r}\right) = S_0\left(\vec{r}\right) - \int_{4\pi} \left(H - H_a\right)\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right)d\Omega$$

or

$$H_a\left[\phi^{l+1}\left(\vec{r}\right) - \tilde{\phi}^{l}\left(\vec{r}\right)\right] = S_0\left(\vec{r}\right) - \int_{4\pi} H\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right)d\Omega \qquad (°)$$

where $\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right)$ is the angular flux obtained by the transport operator making use of the scalar flux obtained at the l-th iteration and included in the scattering term:

$$H_0\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right) = H_1\phi^{l}\left(\vec{r},\vec{\Omega}\right) + \frac{S_0\left(\vec{r}\right)}{4\pi} \qquad (°°)$$

- **We can now note that**

$$H\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right) = H_0\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right) - H_1\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right)$$

- **Making use of (°°), the above becomes**

$$H\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right) = H_1\phi^{l}\left(\vec{r},\vec{\Omega}\right) + \frac{S_0\left(\vec{r}\right)}{4\pi} - H_1\tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right) = H_1\left[\phi^{l}\left(\vec{r},\vec{\Omega}\right) - \tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right)\right] + \frac{S_0\left(\vec{r}\right)}{4\pi}$$

- **Substituting this result into (°), it is found**

$$H_a\left[\phi^{l+1}\left(\vec{r}\right) - \tilde{\phi}^{l}\left(\vec{r}\right)\right] = S_0\left(\vec{r}\right) - \int_{4\pi} \left\{H_1\left[\phi^{l}\left(\vec{r},\vec{\Omega}\right) - \tilde{\phi}^{l}\left(\vec{r},\vec{\Omega}\right)\right] + \frac{S_0\left(\vec{r}\right)}{4\pi}\right\}d\Omega$$

$$H_a\left[\phi^{l+1}(\vec{r})-\tilde{\phi}^{l}(\vec{r})\right] = \int_{4\pi} H_1\left[\tilde{\phi}^{l}(\vec{r},\vec{\Omega})-\phi^{l}(\vec{r},\vec{\Omega})\right]d\Omega$$

- **It can be noted that the above allows to obtain a new guess of the scalar flux operating with the diffusion operator**

$$H_a\cdot = -divD(\vec{r})grad_{\vec{r}}\cdot + \Sigma_r(\vec{r})\cdot \qquad H_1\cdot = \frac{\Sigma_s(\vec{r})}{4\pi}\int_{4\pi}\cdot\, d\Omega'$$

Thus obtaining

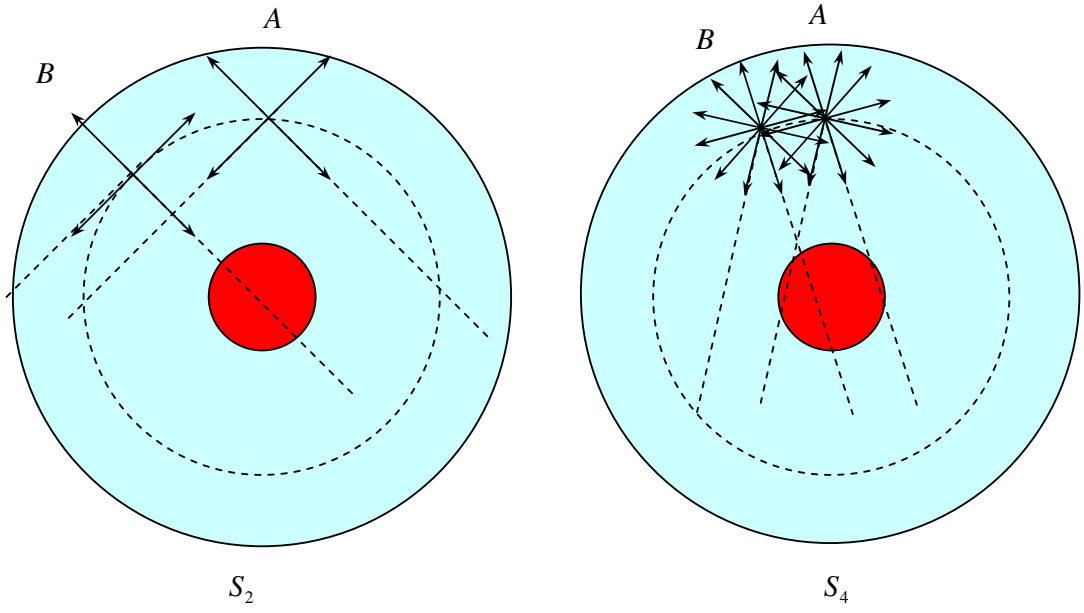
$$\left(div\, D(\vec{r})\, grad_{\vec{r}} - \Sigma_r(\vec{r})\right)\left[\phi^{l+1}(\vec{r})-\tilde{\phi}^{l}(\vec{r})\right] = \Sigma_s(\vec{r})\left[\tilde{\phi}^{l}(\vec{r})-\phi^{l}(\vec{r})\right]$$

- **Therefore, once $\phi^{l}(\vec{r})$ and $\tilde{\phi}^{l}(\vec{r})$ are known, this "easier" formulation allows to update the scalar flux for the next iteration.**

"Ray effects"

- **A classical problem faced by the application of the S_N methods of limited order is the occurrence of oscillations in the computed scalar flux having no physical meaning**
- **The amplitude of such oscillations can be reduced by increasing the order of the S_N method, while their frequency increases**
- **The reason for such behaviour of S_N methods can be considered a direct consequence of the discretization in the angular coordinate**
- **In fact, since the scalar flux is calculated as a weighted average of the angular flux obtained for a limited number of directions, it may happen that its value is perturbed by the discontinuities that the angular flux may show in some particular cases**
- **The Figure below reports the case of a neutron source (the central region) surrounded by a region assumed to be characterised by a scattering macroscopic cross section sufficiently smaller than the total one ($\Sigma_s << \Sigma_t$)**

- **By imposing a boundary condition of free surface on the external boundary, the problem should be typically one-dimensional, and we should expect that on circles (as the dashed one) the scalar flux should be constant**
- **It must be recognised that the above case, chose just for purpose of proposing an example, is very peculiar, because:**
  - **a 1D case should be treated by an appropriate technique taking advantage of one-dimensionality;**
  - **Cartesian coordinates would anyway approximate the circular regions with an irregular boundary.**
- **However, the presented case has the merit to show even more clearly than other examples reported below in the exercises the consequence of a ray effect**
- **In fact, assuming to calculate the flux along the dashed circle in the Figure with the S_2 method and considering that, owing to the relatively large value of the absorption cross section, the flux is mostly made by "first flight" neutrons coming from the source, the resulting scalar flux would result oscillatory, being larger in the location B than in A**
- **This is because the "first flight" neutrons that give a substantial contribution to the scalar flux can hardly reach the location A from**



Typical situation of the occurrence of the "*ray effects*"

the source through the few directions that do not intercept it; viceversa, in the case of point B there is a direction that itercepts the point starting from the source, giving a larger contribution to the angular flux

- **The mitigation of the problem can be obtained by using more many directions, e.g. by an S_4 scheme; as it can be argued from the figure the expected oscillations in the scalar flux will be smaller than with the S_2 method, though their number will increase along the circumference**
- **Another possible solution is to replace the angular discretisation (a sort of angular "collocation") with angular averages over direction intervals. These methods improve the effect at low orders, though they do not completely solve the problem**

GAUSSIAN QUADRATURES

(See Ghelardoni Marzulli "Argomenti di Analisi Numerica" ETS 1980 Vol. II, p. 115 e pp. 132 e sgg.)

Theorem

Considered M distinct values of the abscissa $x_1, x_2, ..., x_M$, in the set of quadratures having the form

$$\int_a^b f(x)dx \approx \sum_{i=1}^{M} w_i f(x_i)$$

there is only one quadrature having exactly the accuracy at least equal to M-1 (i.e, that provides exact integrations of all the polynomials in $[a,b]$ having degree up to M-1)

- **In fact, we need solving the system**

$$\sum_{i=1}^{M} w_i = b - a$$

$$\sum_{i=1}^{M} w_i x_i = \frac{1}{2}\left(b^2 - a^2\right)$$

$$\cdots\cdots$$

$$\sum_{i=1}^{M} w_i x_i^{M-1} = \frac{1}{M}\left(b^M - a^M\right)$$

  whose ("Vandermonde") determinant is certainly non-zero

- **Whenever the $x_1, x_2, ..., x_M$ are not assigned, it is possible to determine the M coefficients w_i and the x_i such that it is possible to integrate exactly polynomials up to the degree 2M-1. We have in fact the equations**

$$\sum_{i=1}^{M} w_i = b - a$$

$$\sum_{i=1}^{M} w_i x_i = \frac{1}{2}\left(b^2 - a^2\right)$$

$$\cdots\cdots$$

$$\sum_{i=1}^{M} w_i x_i^{2M-1} = \frac{1}{2M}\left(b^{2M} - a^{2M}\right)$$

Theorem

If the points $x_1, x_2, ..., x_M$ are the zeroes of the M degree orthogonal polynomials over $[a,b]$ it is possible to construct a quadrature formula

$$\int_a^b f(x)dx \approx \sum_{i=1}^{M} w_i f(x_i)$$

having accuracy of order 2M-1 whose coefficients are the numbers

$$w_i = \int_a^b l_{M-1,i}(x)dx \qquad (i = 1,...,M)$$

with $l_{M-1,i}(x)$ *l'i-th interpolating polynomial having degree M-1*

$$l_{M-1,i}(x) = \frac{(x-x_1)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_M)}{(x_i-x_1)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_M)}$$

- **Such formulations take the name of *Gaussian quadratures***
- **When the interval of definition is $[-1,1]$, the orthogonal polynomials are Legendre polynomials and the above formulations take the name of *Gauss-Legendre quadratures***

EXERCISES WITH AN IN-HOUSE CODE

Solution of the Integro-differential Equation of Neutron Transport in Cartesian 3D Geometry with the Discrete Ordinate Method

1. Description of the Method

The FORTRAN programme is based on the relations already described during lectures for the S_N methods.

In particular:

- the transport equation:

$$div\left[\vec{\Omega}\phi\left(\vec{r},\vec{\Omega}\right)\right] + \Sigma_t\left(\vec{r}\right)\phi\left(\vec{r},\vec{\Omega}\right) - q\left(\vec{r},\vec{\Omega}\right) = 0$$

is spatially discretised as

$$\mu_m \frac{\phi_{i+1/2\,jk,m} - \phi_{i-1/2\,jk,m}}{\Delta x_i} + \eta_m \frac{\phi_{i,j+1/2k,m} - \phi_{i,j-1/2k,m}}{\Delta y_j} + \zeta_m \frac{\phi_{ijk+1/2,m} - \phi_{ijk-1/2,m}}{\Delta z_k}$$
$$+ \Sigma_{t,ijk}\phi_{ijk,m} = q_{ijk,m} \qquad (i=1,...,I);(j=1,...,J);(k=1,...,K);(m=1,...,N_m)$$

- the "diamond rule" is applied coherently with the sign of direction cosines:

$$\begin{cases} \mu_m > 0 & \phi_{i+1/2,\,jk,m} = 2\phi_{ijk,m} - \phi_{i-1/2,\,jk,m} \\ \mu_m < 0 & \phi_{i-1/2,\,jk,m} = 2\phi_{ijk,m} - \phi_{i+1/2,\,jk,m} \end{cases}$$
$$\begin{cases} \eta_m > 0 & \phi_{i,\,j+1/2,k,m} = 2\phi_{ijk,m} - \phi_{i,\,j-1/2,k,m} \\ \eta_m < 0 & \phi_{i,\,j-1/2,k,m} = 2\phi_{ijk,m} - \phi_{i,\,j+1/2,k,m} \end{cases}$$
$$\begin{cases} \zeta_m > 0 & \phi_{ij,k+1/2,m} = 2\phi_{ijk,m} - \phi_{ij,k-1/2,m} \\ \zeta_m < 0 & \phi_{ij,k-1/2,m} = 2\phi_{ijk,m} - \phi_{ij,k+1/2,m} \end{cases}$$

obtaining formulations having the form:

- $\mu_m > 0, \eta_m > 0, \zeta_m > 0$

$$\phi_{ijk,m} = \frac{\frac{2\mu_m}{\Delta x_i}\phi_{i-1/2\,jk,m} + \frac{2\eta_m}{\Delta y_j}\phi_{ij-1/2k,m} + \frac{2\zeta_m}{\Delta z_k}\phi_{ijk-1/2,m} + q_{ijk,m}}{\frac{2\mu_m}{\Delta x_i} + \frac{2\eta_m}{\Delta y_j} + \frac{2\zeta_m}{\Delta z_k} + \Sigma_{t,ijk}}$$

- $\mu_m < 0, \eta_m > 0, \zeta_m < 0$

$$\phi_{ijk,m} = \frac{-\frac{2\mu_m}{\Delta x_i}\phi_{i+1/2\,jk,m} + \frac{2\eta_m}{\Delta y_j}\phi_{ij-1/2k,m} - \frac{2\zeta_m}{\Delta z_k}\phi_{ijk+1/2,m} + q_{ijk,m}}{-\frac{2\mu_m}{\Delta x_i} + \frac{2\eta_m}{\Delta y_j} - \frac{2\zeta_m}{\Delta z_k} + \Sigma_{t,ijk}}$$

- The "step" rule for fix-up" is used when

$$\phi_{i+1/2,jk,m} < 0 \qquad \text{with} \quad \mu_m > 0, \eta_m > 0, \zeta_m > 0 \qquad ;$$

in place of the diamond rule, it is assumed

$$\phi_{i+1/2,jk,m} = \phi_{ijk,m}$$

and the average flux is obtained as

$$\phi_{ijk,m} = \frac{\frac{\mu_m}{\Delta x_i}\phi_{i-1/2\,jk,m} + \frac{2\eta_m}{\Delta y_j}\phi_{ij-1/2k,m} + \frac{2\zeta_m}{\Delta z_k}\phi_{ijk-1/2,m} + q_{ijk,m}}{\frac{\mu_m}{\Delta x_i} + \frac{2\eta_m}{\Delta y_j} + \frac{2\zeta_m}{\Delta z_k} + \Sigma_{t,ijk}} = \phi_{i+1/2,jk,m}$$

- LQ_n quadratures are used for the admisssible directions and their weights (up to S_8)

S_2: 1

S_4: 1 / 1 1

S_6: 1 / 2 2 / 1 2 1

S_8: 1 / 2 2 / 2 3 2 / 1 2 2 1

S_{12}: 1 / 2 2 / 3 4 3 / 3 5 5 3 / 2 4 5 4 2 / 1 2 3 3 2 1

S_{16}: 1 / 2 2 / 3 5 3 / 4 6 6 4 / 4 7 8 7 4 / 3 6 8 8 6 3 / 2 5 6 7 6 5 2 / 1 2 3 4 4 3 2 1

Level	n	μ_n	w_n^h
S_4	1	0.3500212	0.3333333
	2	0.8688903	
S_6	1	0.2666355	0.1761263
	2	0.6815076	0.1572071
	3	0.9261808	
S_8	1	0.2182179	0.1209877
	2	0.5773503	0.0907407
	3	0.7867958	0.0925926
	4	0.9511897	
S_{12}	1	0.1672126	0.0707626
	2	0.4595476	0.0558811
	3	0.6280191	0.0373377
	4	0.7600210	0.0502819
	5	0.8722706	0.0258513
	6	0.9716377	
S_{16}	1	0.1389568	0.0489872
	2	0.3922893	0.0413296
	3	0.5370966	0.0212326
	4	0.6504264	0.0256207
	5	0.7467506	0.0360486
	6	0.8319966	0.0144589
	7	0.9092855	0.0344958
	8	0.9805009	0.0085179

(from E.E. Lewis and W.F. Miller, Jr. "Computational Methods of Neutron Transport", John Wiley and Sons, 1984)

In particular:

- the directions are generated in the octant with $N(N+2)/8$ positive direction cosines, obtaining those in the othe octants by rotation around the Cartesian axes;
- given from the previous table the value of μ_1, the other values are obtained by:

$$\mu_i^2 = \mu_1^2 + (i-1)C \qquad C = \frac{2\left(1-3\mu_1^2\right)}{N-2}$$

- the weighting factors are obtained by the previous tables, where the last table specifies the weighting factors for directions labeled by a given value of *n* in the first table
- in the aim to minimize memory use, the value of the angular flux for any direction is not stored in memory, assigning its contribution to the scalar fux in an incremental way for each direction

- the scalar flux is calculated as:

$$\phi_{ijk} = \frac{1}{8}\sum_{m=1}^{N(N+2)} w_m \phi_{ijk,m}$$

where the weighting coefficients are normalized over an octant

- Taking into account this formulation, the emission density must be properly evaluated; in fact, multiplying by $\Delta x_i \Delta y_j \Delta z_k \; w_m/8$ boths sides of the relationship

$$\mu_m \frac{\phi_{i+1/2\,jk,m} - \phi_{i-1/2\,jk,m}}{\Delta x_i} + \eta_m \frac{\phi_{i,j+1/2k,m} - \phi_{i,j-1/2k,m}}{\Delta y_j} + \zeta_m \frac{\phi_{ijk+1/2,m} - \phi_{ijk-1/2,m}}{\Delta z_k} + \Sigma_{t,ijk}\phi_{ijk,m} = q_{ijk,m}$$

and taking the summation over *m*, it is

$$\frac{1}{8}\left\{\sum_{m=1}^{N(N+2)} w_m \mu_m \phi_{i+1/2\,jk,m} - \sum_{m=1}^{N(N+2)} w_m \mu_m \phi_{i-1/2\,jk,m}\right\}\Delta y_j \Delta z_k$$

$$+\frac{1}{8}\left\{\sum_{m=1}^{N(N+2)} w_m \eta_m \phi_{ij+1/2k,m} - \sum_{m=1}^{N(N+2)} w_m \eta_m \phi_{ij-1/2k,m}\right\}\Delta x_i \Delta z_k$$

$$+\frac{1}{8}\left\{\sum_{m=1}^{N(N+2)} w_m \zeta_m \phi_{ijk+1/2,m} - \sum_{m=1}^{N(N+2)} w_m \zeta_m \phi_{ijk-1/2,m}\right\}\Delta x_i \Delta y_j$$

$$+\Sigma_{t,ijk}\frac{1}{8}\sum_{m=1}^{N(N+2)} w_m \phi_{ijk,m}\Delta x_i \Delta y_j \Delta z_k = \frac{1}{8}\sum_{m=1}^{N(N+2)} w_m q_{ijk,m}\Delta x_i \Delta y_j \Delta z_k$$

and then

$$\left\{\vec{J}_{i+1/2\,jk}\cdot\vec{i}-\vec{J}_{i-1/2\,jk}\cdot\vec{i}\right\}\Delta y_j\Delta z_k+\left\{\vec{J}_{ij+1/2k}\cdot\vec{j}-\vec{J}_{ij-1/2k}\cdot\vec{j}\right\}\Delta x_i\Delta z_k$$
$$+\left\{\vec{J}_{ijk+1/2}\cdot\vec{k}-\vec{J}_{ijk-1/2}\cdot\vec{k}\right\}\Delta x_i\Delta y_j+\Sigma_{t,ijk}\phi_{ijk}\Delta x_i\Delta y_j\Delta z_k=q_{ijk}\Delta x_i\Delta y_j\Delta z_k$$

where $\vec{i}$, $\vec{j}$ e $\vec{k}$ represent the unit vectors parallel to the Cartesian axes and it is assumed

$$\vec{J}_{i\pm1/2\,jk}\cdot\vec{i}=\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m\mu_m\phi_{i\pm1/2\,jk,m}$$

$$\vec{J}_{ij\pm1/2k}\cdot\vec{j}=\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m\eta_m\phi_{ij\pm1/2k,m}$$

$$\vec{J}_{ijk\pm1/2}\cdot\vec{k}=\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m\zeta_m\phi_{ijk\pm1/2,m}$$

$$q_{ijk}=\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m q_{ijk,m}$$

The obtained balance equation obviously represents the neutron balance on the volume.

For the emission density, it is noted that in the case of isotropic scattering and independent source it is:

$$q_{ijk,m}=\overline{q}_{ijk}\qquad\Rightarrow\qquad q_{ijk}=\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m q_{ijk,m}=\overline{q}_{ijk}$$

where use is made of the relationship

$$\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m=1$$

In summary, it is necessary to remember that in the definition of the emission density the weighting factors are normalised to one instead of 4π.

For isotropic sources, it is:

$$q_{ijk}=\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m\left\{\Sigma_s\phi_{ijk,m}+s_{ijk,m}\right\}=\Sigma_s\frac{1}{8}\sum_{m=1}^{N(N+2)}w_m\phi_{ijk,m}+s_{ijk}$$

and then

$$q_{ijk,m} = \Sigma_s \phi_{ijk} + s_{ijk}$$

Finally, since the emission density contains the scattring source, it is possible to iterate on the flux starting from an estimate of the emission density based on the independent source only:

$$q^0_{ijk,m} = s_{ijk}$$

2. *Structure of the program input file*

An example of input file is reported hereafter.

```
Number of x layers     Number of y layers    Number of z layers         Order of the SN Method
             1                      1                     1                              6
       x Layer Thickness      Number of nodes in each x layer (repeat for each layer)
                      4.                          40
       y Layer Thickness      Number of nodes in each y layer (repeat for each layer)
                      4.                          40
       z Layer Thickness      Number of nodes in each z layer (repeat for each layer)
                     10.                        40
   Sigma Tot. in general    Sigma Scatt. in general     Source in general
              0.750000                 0.500000                    0.
       Number of regions with properties different from the general ones
                                                                  4
       Characteristics of the regions:
        x1      x2      y1     y2      z1     z2   SigmaTot.    SigmaScatt.    Source
       0.5     1.5     1.5    2.5     2.     8.    0.750000       0.500000           1.
       1.5     2.5     0.5    1.5     2.     8.    0.750000       0.500000           1.
       2.5     3.5     1.5    2.5     2.     8.    0.750000       0.500000           1.
       1.5     2.5     2.5    3.5     2.     8.    0.750000       0.500000           1.
   Boundary conditions: value of the flux on the six lateral surfaces
    West Face     East Face     South Face     North Face     Bottom Face    Top Face
            0.            0.             0.             0.              0.           0.
   Max. Error on Flux
              1.d-05
```

As it can be noted, the programme allows defining different discretization steps in different layers along the three axes.

General nuclear properties are assigned for the material in the integration domain, though in parallelepipedal regions it is possible to specify different values

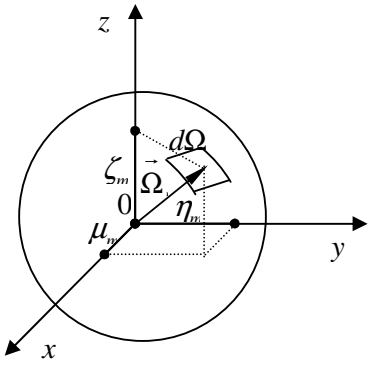
for the cross sections and the source. In this way, it is possible to specify problems characterized by different sources and property distributions.

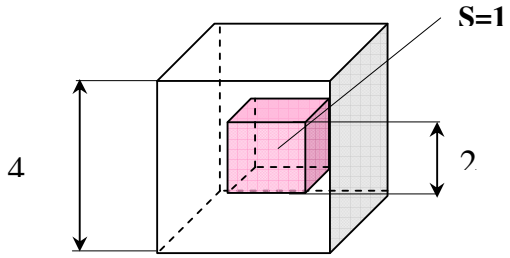
It is possible to assign over the six lateral surfaces boundary conditions in terms of inward angular fluxes (isotropic for the inward directions). Imposing that these fluxes are zero, an isolated body is considered. By convention, assigning negative fluxes, pure reflective conditions are assumed by the code.

3. First Applications

Isolated Cube Made of Absorbing Material

with Internal Source in a Cubical Region





$\Sigma_t = 3/4$

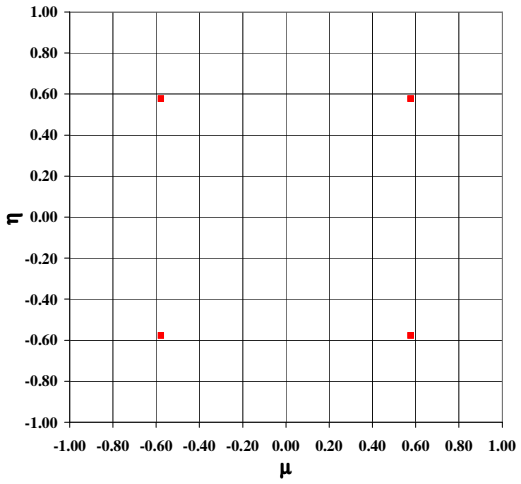
$\Sigma_s = 2/3\Sigma_t$

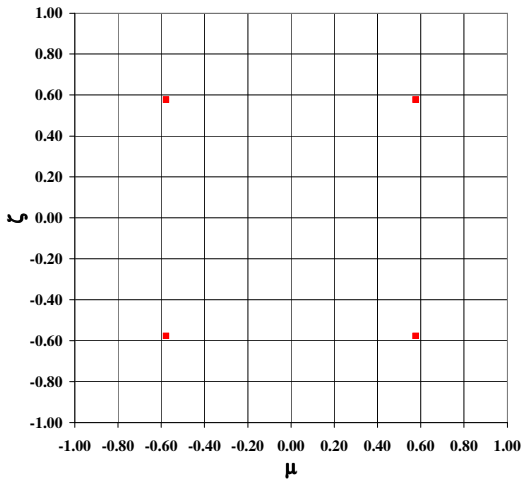
- Input deck

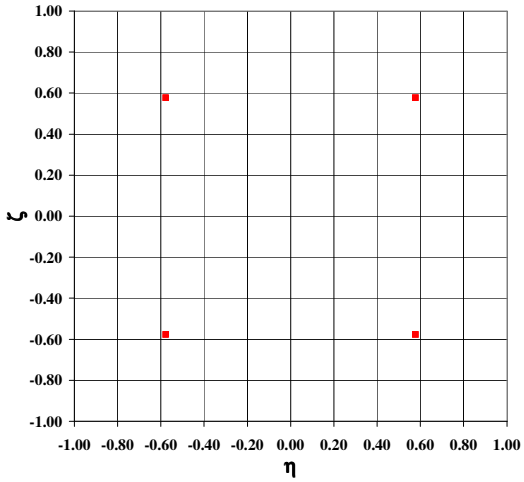
```
Number of x layers     Number of y layers    Number of z layers         Order of the SN Method
             1                      1                     1                              2
       x Layer Thickness      Number of nodes in each x layer (repeat for each layer)
                    4.                         40
       y Layer Thickness      Number of nodes in each y layer (repeat for each layer)
                    4.                         40
       z Layer Thickness      Number of nodes in each z layer (repeat for each layer)
                    4.                         40
   Sigma Tot. in general    Sigma Scatt. in general     Source in general
                0.750000                 0.500000                    0.
       Number of regions with properties different from the general ones
                                                                    1
       Characteristics of the regions:
       x1      x2      y1     y2      z1     z2   SigmaTot.    SigmaScatt.    Source
       1.      3.      1.     3.      1.     3.   0.750000       0.500000           1.
   Boundary conditions: value of the flux on the six lateral surfaces
    West Face     East Face     South Face     North Face     Bottom Face    Top Face
            0.            0.             0.             0.             0.           0.
   Max. Error on Flux
             1.d-05
```

- Check on the distribution of the discrete ordinates for the different orders of the method

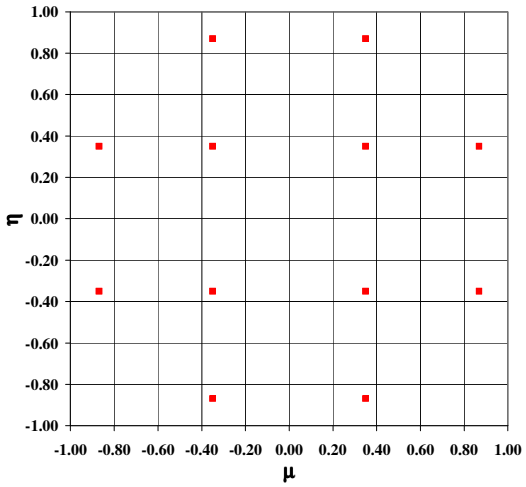
S_2 approximation

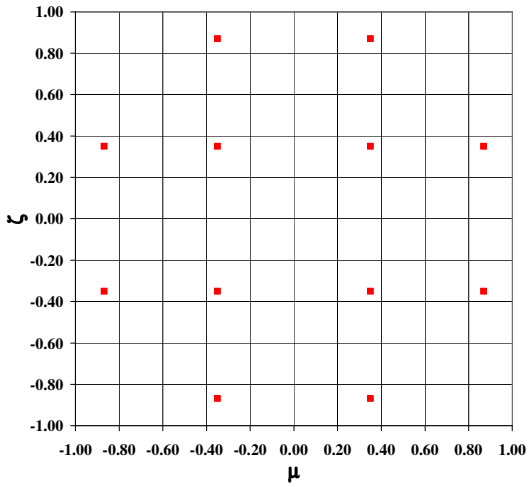


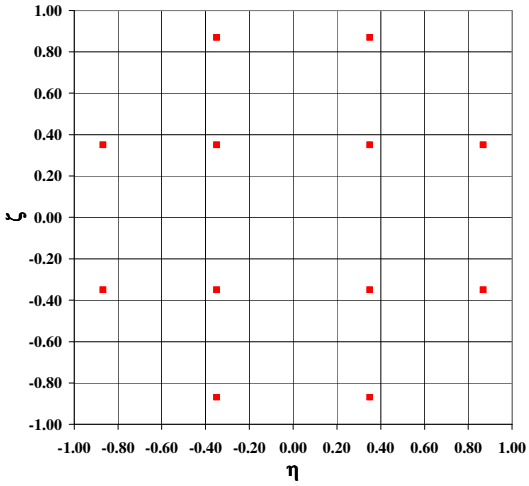




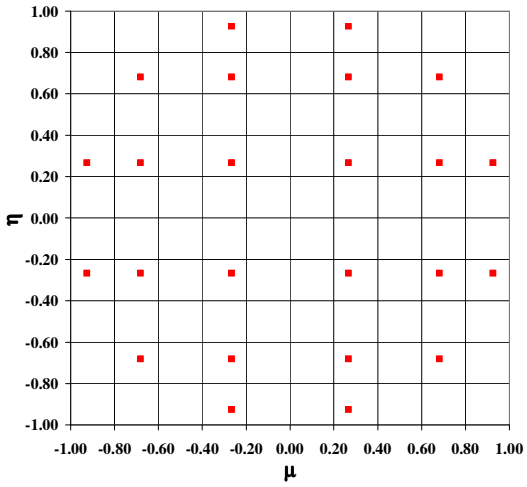
S_4 approximation

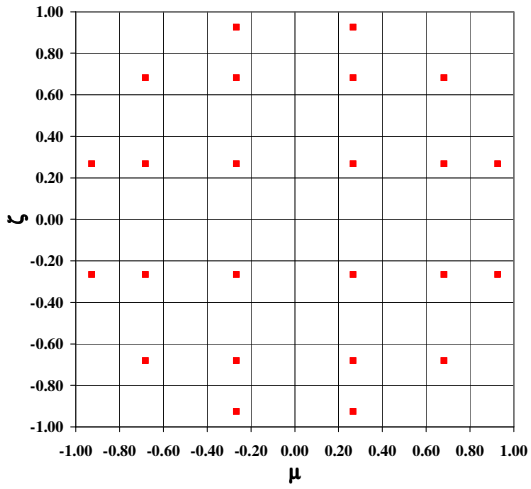


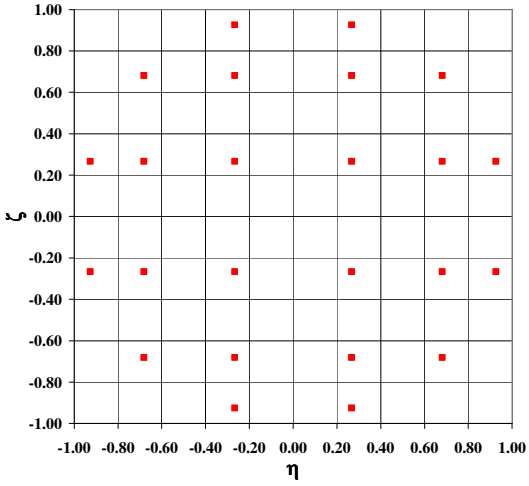




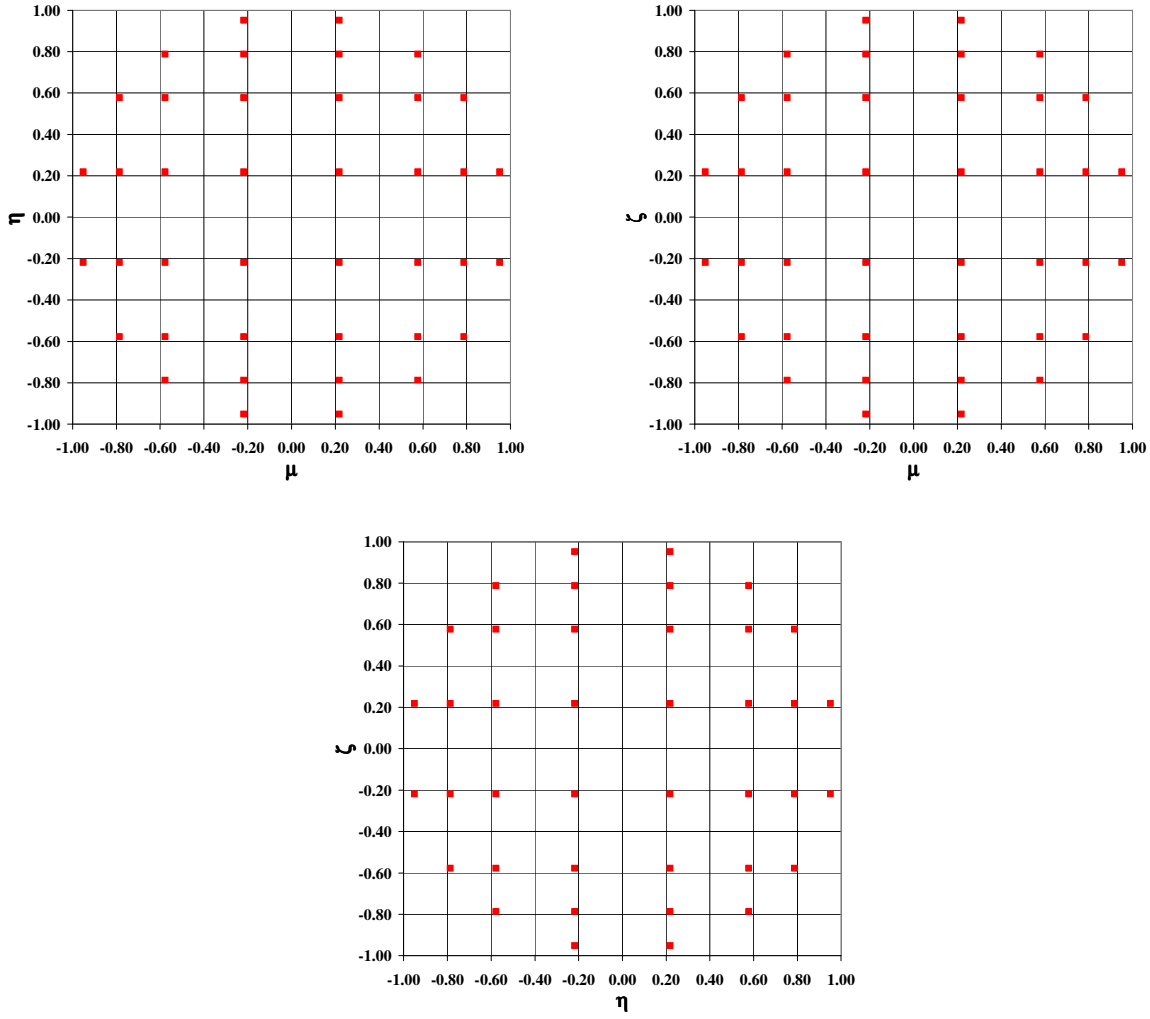
S_6 approximation





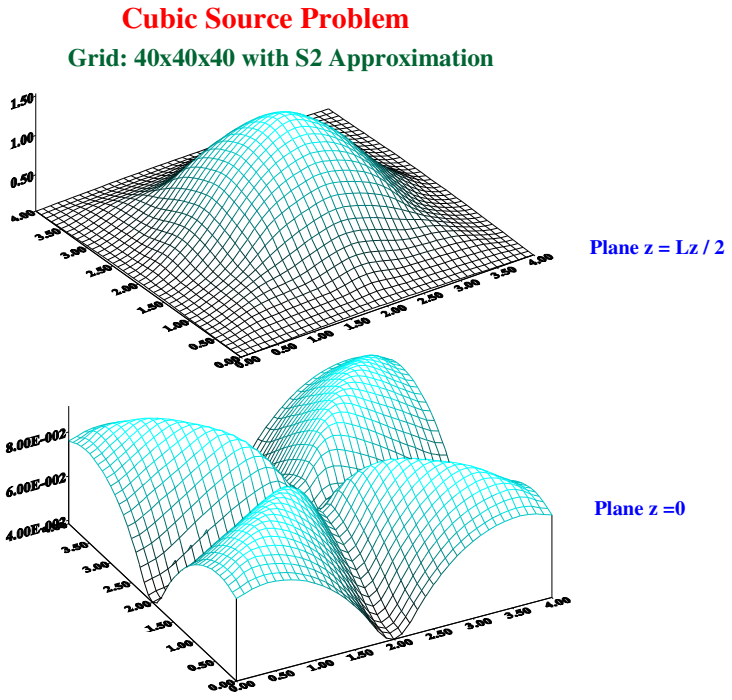


S_8 approximation



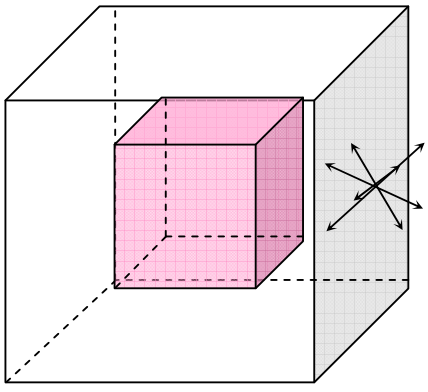
- It can be noted that:
  - Increasing the order of the approximation, the unity radius sphere is gradually populated by directions;
  - the directions are the same for ±90° rotations with respect to any frame axis.

Results:



The explanation of the behaviour observed on the lateral surfaces becomes evident considering that the adimissible directions are in a too small number to provide a sufficiently detailed representation of the scalar flux.

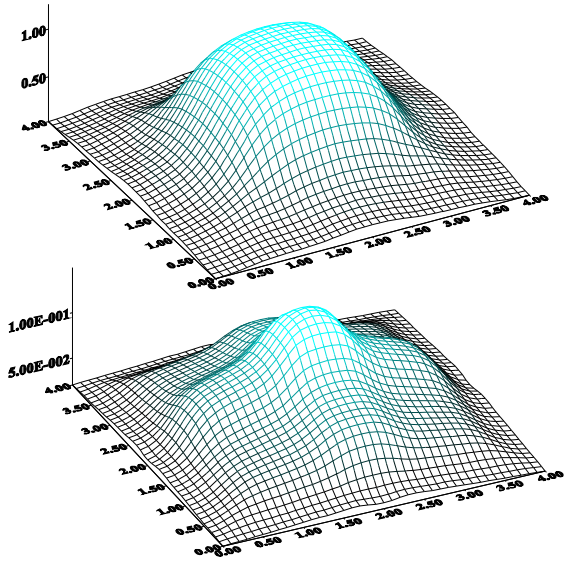
For instance, in the position illustrated in the following figure the 8 directions do not intersect the source, resulting in a minimum of the scalar flux.



This occurs with perfect symmetry for any lateral surface.

Cubic Source Problem

Grid: 40x40x40 with S4 Approximation

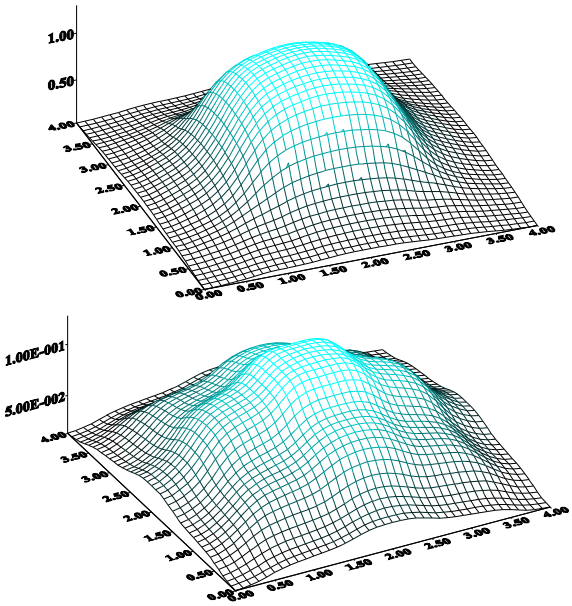


Plane z = Lz / 2

Plane z =0

Cubic Source Problem

Grid: 40x40x40 with S6 Approximation

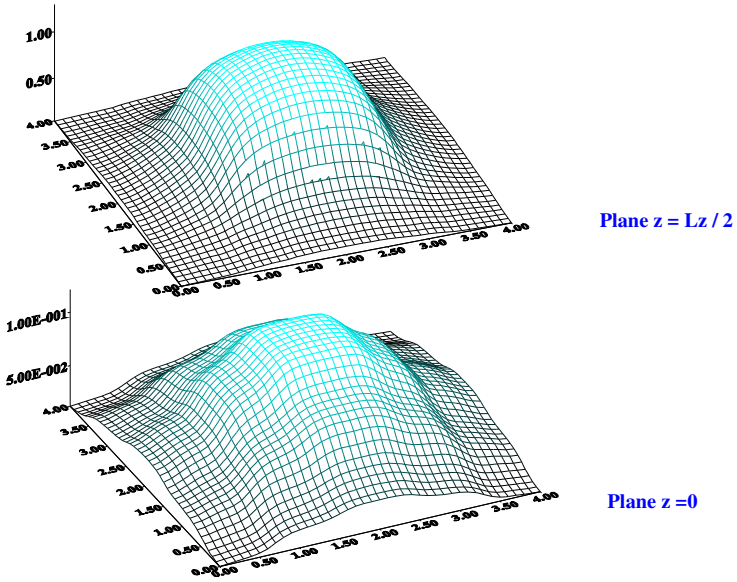


Plane z = Lz / 2

Plane z =0

Cubic Source Problem

Grid: 40x40x40 with S8 Approximation



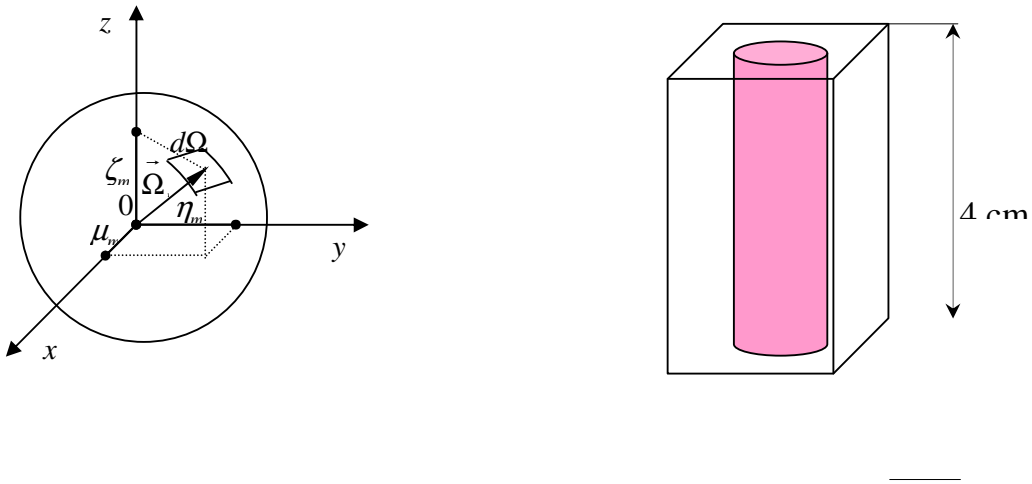
Plane z = Lz / 2

Plane z =0

- The "ray effect" is particularly clear in the case of the S_2 approximation, that provides a totally unrealistic trend of the scalar flux on the plane z = 0, to be considered in view of the only eight admissible directaion (one per each octant).

- By increasing the number of directions, the problem is mitigated, though oscillatory trends are anyway observed, to be put in relation with ray effects.

Square Cell with Cylindrical Rod



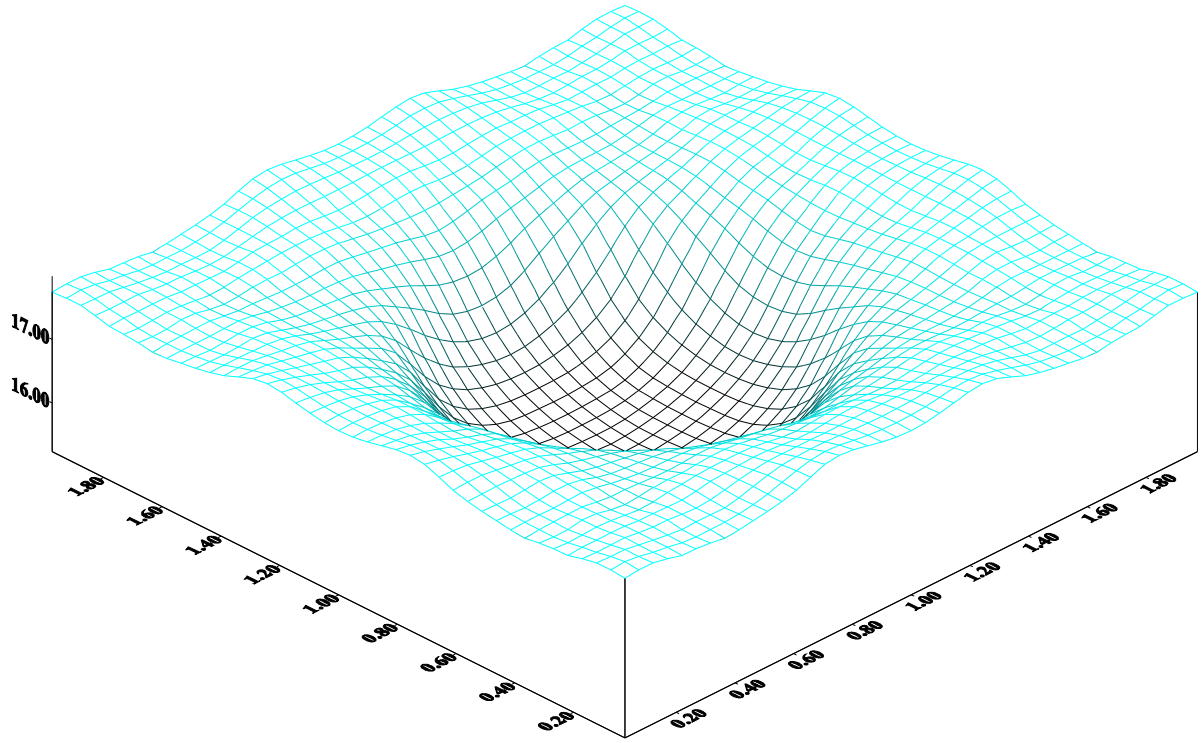
The calculation is performed with the S8 approximation, making use of the following input file, where the absolute of the negative value of y2 indicates radius of the rode centered in (x1,y1).

```
Number of x layers     Number of y layers    Number of z layers         Order of the SN Method
             1                      1                     1                              8
      x Layer Thickness      Number of nodes in each x layer (repeat for each layer)
                     2.                          40
      y Layer Thickness      Number of nodes in each y layer (repeat for each layer)
                     2.                          40
      z Layer Thickness      Number of nodes in each z layer (repeat for each layer)
                     4.                          40
   Sigma Tot. in general    Sigma Scatt. in general     Source in general
                 0.50000                 0.500000                    1.
      Number of regions with properties different from the general ones
                                                                    1
      Characteristics of the regions:
        x1      x2      y1      y2      z1      z2    SigmaTot.    SigmaScatt.    Source
        1.    -0.5      1.      0.      0.      4.    0.750000       0.500000           0.
      Boundary conditions: value of the flux on the six lateral surfaces
       West Face     East Face     South Face     North Face     Bottom Face    Top Face
              -1.           -1.            -1.            -1.             -1.          -1.
      Max. Error on Flux
      1.d-05
```

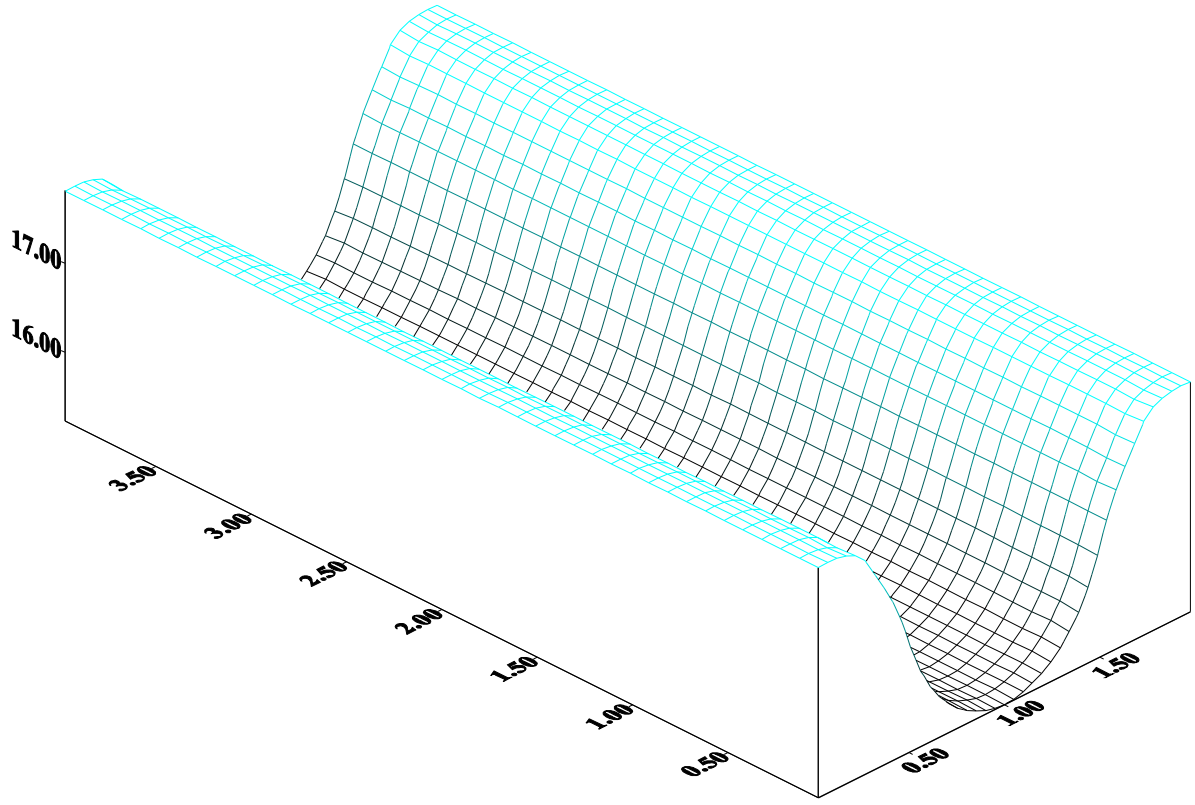
Note that the values of the cross sections are purely parametric.

The results show a classical spatial trend of the thermal flux (the source is located in the moderator). A considerable depression of the flux is noted in the rod in the plane $z = L_z/2$. Flux oscillations due to the "ray effects" are again observed.

It is also interesting to note that that along an axial plane the flus is absolutely independent from the z coordinate, testifying for the effectiveness of the choice of reflective boundary conditions on the top and bottom planes in transforming the 3D problem into a 2D one.



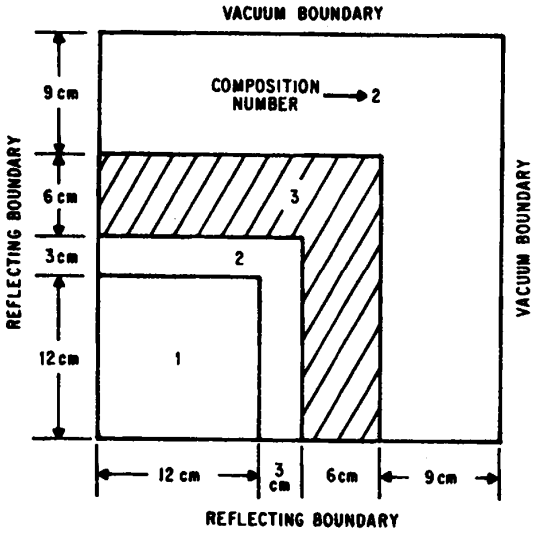
Plane z = 2 cm



Plane y = 0

Benchmark Problem Iron-Water

(v. H. Khalil, *Nucl. Sci. Eng.*, 90, pp.263-280, 1985)



COMPOSITION	Σ_t (cm^{-1})	C	SOURCE STRENGTH ($cm^{-3} \cdot s^{-1}$)
1 (WATER)	3.33	0.994	1.0
2 (WATER)	3.33	0.994	0.0
3 (IRON)	1.33	0.831	0.0

Fig. 2. Geometry and material properties for the iron-water benchmark problem.

TABLE IV

Results of the Iron-Water Benchmark Problem*

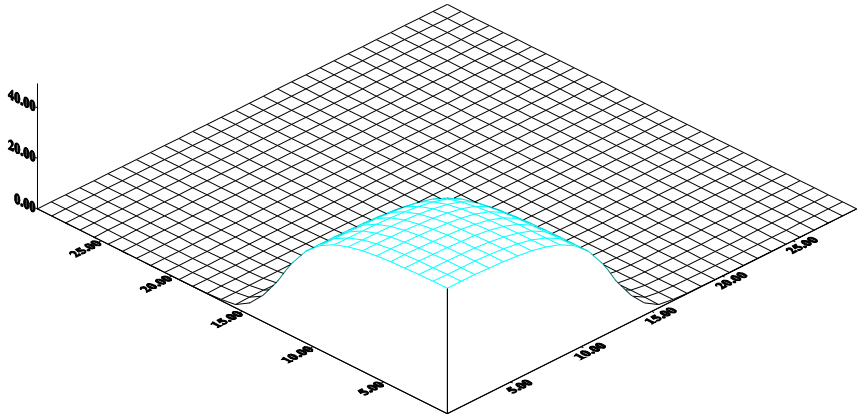
Number of X-Y mesh cells, $I \times J$	6×6	10×10	20×20
Maximum τ_u	20.0	10.0	5.0
Minimum τ_u	4.0	4.0	2.0
Number of synthetic iterations	4	4	4
Computing time for diffusion iterations (s)	0.18	0.39	1.73
Computing time for transport sweeps (s)	0.06	0.14	0.51
Total solution time (s)	0.24	0.53	2.24
Average flux by composition[a] ($cm^{-2} \cdot s^{-1}$)			
1	4.0060+1[b]	4.0652+1	4.0900+1
2	1.5896+0	1.5259+0	1.4164+0
3	2.3885−1	2.1782−1	2.2744−1

*Directional approximation = S_4.
[a]See Fig. 2.
[b]Read as 4.0060×10^1.

The problem is simulated in 3D version with reflective conditions on the faces orthogonal to the z axis. The input file is the following

```
Number of x layers     Number of y layers    Number of z layers         Order of the SN Method
             1                      1                     1                              8
      x Layer Thickness      Number of nodes in each x layer (repeat for each layer)
                    30.                          30
      y Layer Thickness      Number of nodes in each y layer (repeat for each layer)
                    30.                          30
      z Layer Thickness      Number of nodes in each z layer (repeat for each layer)
                    10.                          10
   Sigma Tot. in general    Sigma Scatt. in general     Source in general
                    3.33                         3.31                    0.
      Number of regions with properties different from the general ones
                                                                         3
      Characteristics of the regions:
```

```
x1      x2      y1      y2      z1      z2    SigmaTot.    SigmaScatt.    Source
0.      12.      0.     12.      0.     10.        3.33          3.31           1.
15.     21.      0.     21.      0.     10.        1.33          1.105          0.
0.      21.     15.     21.      0.     10.        1.33          1.105          0.
Boundary conditions: value of the flux on the six lateral surfaces
 West Face     East Face     South Face     North Face     Bottom Face    Top Face
       -40.            0.            -40.            0.             -40.        -40.
Max. Error on Flux
1.d-05
```



It can be noted that the average value of the flux in region 1 is 4.09e01, in in agreement with tabulated data.

4. Proposed activities

- Analyse the computer program trying to indentify the following phases of the calculation:
  - assignement of directions and of the related weights;
  - solution of the transport equation in agreement with the sign of direction cosines;
  - updating the emission density.

```
c-----------------------------------------------------------------------c
c                                                                       c
c    Program for the solution of the integrodifferential equation       c
c    for transport in 3D geometry                                       c
c                                                                       c
c    This programme has been set up for teaching purposes               c
c    and was not subjected to a sifficiently thorough validation        c
c    to assure the quantitative correctness of the results              c
c                                                                       c
c    W. Ambrosini, DIMNP, Dicember 2000                                 c
c                                                                       c
c-----------------------------------------------------------------------c
      program tras3d
      implicit double precision (a-h,o-z)
c
      include 'tras3dc.for'
c
      open (unit=5,file='schermo.dat')
      open (unit=6,file='tras3d.out')
      open (unit=7,file='tras3d.ord')
        open (unit=8,file='trasxy.txt')
        open (unit=9,file='trasxz.txt')
        open (unit=10,file='trasyz.txt')
        open (unit=11,file='trasxy1.txt')
        open (unit=12,file='trasxz1.txt')
        open (unit=13,file='trasyz1.txt')
        open (unit=14,file='trasxy2.txt')
        open (unit=15,file='trasxz2.txt')
        open (unit=16,file='trasyz2.txt')
c
c   Pi is calculaetd
c
      pi = 4.d00 * datan (1.d00)
c
c   Reading the number of layers along the thtee axes and of the order of the method
c
      read (5,*)
        read (5,*) nlax,nlay,nlaz,nordin
c
c   Reading of the thickness of the layer and assignment of the node coordinates (x)
c
      read (5,*)
c
      x(1) = 0.d00
        i = 1
c
          do ilax = 1,nlax
          read (5,*) deltx,nlx
        dx = deltx / dfloat (nlx)
c
           do ix = 1,nlx
           i = i + 1
             x(i) = x(i-1) + dx
             xm(i-1) = x(i-1) + 0.5d00 * dx
           enddo
c
        enddo
c
      nx = i - 1
c
c   Reading of the thickness of the layer and assignment of the node coordinates (y)
c
      read (5,*)
c
      y(1) = 0.d00
        j = 1
c
          do ilay = 1,nlay
          read (5,*) delty,nly
        dy = delty / dfloat (nly)
c
           do iy = 1,nly
           j = j + 1
             y(j) = y(j-1) + dy
             ym(j-1) = y(j-1) + 0.5d00 * dy
           enddo
c
        enddo
c
      ny = j - 1
c
c   Reading of the thickness of the layer and assignment of the node coordinates (z)
c
      read (5,*)
c
      z(1) = 0.d00
        k = 1
c
          do ilaz = 1,nlaz
          read (5,*) deltz,nlz
        dz = deltz / dfloat (nlz)
```

```
c
            do iz = 1,nlz
            k = k + 1
               z(k) = z(k-1) + dz
               zm(k-1) = z(k-1) + 0.5d00 * dz
            enddo
c
          enddo
c
        nz = k - 1
c
c   Reading of cross sections and of the source for general nodes
c
        read (5,*)
          read (5,*) sigmat,sigmas,sourt
c
          do i = 1,nx
              do j = 1,ny
               do k = 1,nz
                sigt(i,j,k) = sigmat
                sigs(i,j,k) = sigmas
                sour(i,j,k) = sourt
               enddo
              enddo
             enddo
c
c   Reading of the number and of the position of the regions having
c   properties different from the general ones
c
        read (5,*)
          read (5,*) nreg
c
        read (5,*)
        read (5,*)
c
             do ir = 1,nreg
             read (5,*) x1,x2,y1,y2,z1,z2,sigmat,sigmas,sval
c
             if(x2.gt.0.d00) then
              do i = 1,nx
                      do j = 1,ny
                       do k = 1,nz
                            if( (xm(i).ge.x1).and.(xm(i).le.x2)
     &                .and.(ym(j).ge.y1).and.(ym(j).le.y2)
     &                  .and.(zm(k).ge.z1).and.(zm(k).le.z2) ) then
                     sour(i,j,k) = sval
                     sigt(i,j,k) = sigmat
                     sigs(i,j,k) = sigmas
                     endif
                  enddo
                   enddo
                  enddo
c
             else
             radius = dabs ( x2 )
c
              do i = 1,nx
                 distx = xm(i) - x1
c
                      do j = 1,ny
                  disty = ym(j) - y1
                  rad = dsqrt ( distx * distx + disty * disty )
c
                       do k = 1,nz
                            if( (rad.le.radius)
     &                  .and.(zm(k).ge.z1).and.(zm(k).le.z2) ) then
                    sour(i,j,k) = sval
                    sigt(i,j,k) = sigmat
                    sigs(i,j,k) = sigmas
                    endif
                 enddo
                  enddo
                 enddo
c
               endif
c
             enddo
c
c   Reading of the inward fluxes at the six lateral surfaces
c
        read (5,*)
        read (5,*)
          read (5,*) phiawf,phiaef,phiasf,phianf,phiabf,phiatf
c
c   Definition of the maximum erro between two subsequent iterations
c
        read (5,*)
          read (5,*) epsphi
c
c   Assignmenet of the discrete ordinates
c
```

```
      call diradm
c
c  Assignement of the initial value of the source and of the fluxes at previous iteration
c
        do i = 1,nx
            do j = 1,ny
             do k = 1,nz
              q(i,j,k) = sour(i,j,k)
c
              oldphi(i,j,k) = 0.d00
c
             enddo
            enddo
           enddo
c
        do im = 1,nm
c
         do j = 1,ny
          do k = 1,nz
          phiaem(j,k,im) = dabs (phiaef)
          phiawm(j,k,im) = dabs (phiawf)
            enddo
           enddo
c
         do i = 1,nx
          do k = 1,nz
          phiasm(i,k,im) = dabs (phiasf)
          phianm(i,k,im) = dabs (phianf)
            enddo
           enddo
c
         do i = 1,nx
          do j = 1,ny
          phiabm(i,j,im) = dabs (phiabf)
          phiatm(i,j,im) = dabs (phiatf)
            enddo
           enddo
c
          enddo
c
c  Loop for iteration on te scattering source
c
        do iscat = 1,10000000
c
        call trasp
c
        call sscat
c
c  Cechk on convergence on flux
c
        dphimx = 0.d00
c
           imax = 1
           jmax = 1
           kmax = 1
c
            do i = 1,nx
             do j = 1,ny
              do k = 1,nz
c
            absdif = dabs ( phi(i,j,k) - oldphi(i,j,k) )
c
                  if(dphimx.lt.absdif) then
                  dphimx = absdif
              imax = i
              jmax = j
                   kmax = k
              endif
c
              enddo
             enddo
            enddo
c
        write(*,100) iscat,dphimx
          write(*,*) imax,jmax,kmax
          write(*,*) phi(imax,jmax,kmax),oldphi(imax,jmax,kmax)
        write(6,100) iscat,dphimx
c
c   Criterion for stopping the iterations
c
        if(dphimx.lt.epsphi) goto 10
c
c   Assigning the flux at the old iterations
c
        do i = 1,nx
            do j = 1,ny
             do k = 1,nz
c
              oldphi(i,j,k) = phi(i,j,k)
c
             enddo
```

```
          enddo
         enddo
c
       enddo
c
   10 continue
c
       nxh = nx / 2
       nyh = ny / 2
       nzh = nz / 2
c
c   Mapa for the planes z = zmax / 2 and z = 0
c
       do i = 1,nx
          do j = 1,ny
c
         write(8,200) xm(i),ym(j),phi(i,j,nzh)
         write(11,200) xm(i),ym(j),phi(i,j,1)
         write(14,200) xm(i),ym(j),phi(i,j,nz)
c
          enddo
         enddo
c
c   Maps for the plane y = ymax / 2 and y = 0
c
       do i = 1,nx
          do k = 1,nz
c
         write(9,200) xm(i),zm(k),phi(i,nyh,k)
         write(12,200) xm(i),zm(k),phi(i,1,k)
         write(15,200) xm(i),zm(k),phi(i,ny,k)
c
          enddo
         enddo
c
c   Maps for the plane x = xmax / 2 and x = 0
c
       do j = 1,ny
          do k = 1,nz
c
         write(10,200) ym(j),zm(k),phi(nxh,j,k)
         write(13,200) ym(j),zm(k),phi(1,j,k)
         write(16,200) ym(j),zm(k),phi(nx,j,k)
c
          enddo
         enddo
c
      stop
  100 format(/,1x,' Iterazione sulla Sorgente di Scattering N. ',i6,
     &          ' Errphi = ',e14.7,/)
  200 format(3(1x,e14.7))
       end
c-----------------------------------------------------------------------c
c                                                                       c
c    Subroutine generating admissible directins in an octant            c
c                                                                       c
c-----------------------------------------------------------------------c
      subroutine diradm
      implicit double precision (a-h,o-z)
c
      include 'tras3dc.for'
c
c
c   Assignement of the discrete ordinetes and of their weights
c
      nm = nordin * ( nordin + 2 )
      noct = nm / 8
c
c   Parameters for the quadrature LQ2
c
       if(nordin.eq.2) then
c
         rsqr3 = 1.d00 / dsqrt(3.d00)
c
         ami(1) = rsqr3
         eta(1) = rsqr3
         zeta(1) = rsqr3
c
         w(1) = 1.d00
c
         call rotoct (1)
c
c   Parameters for the quadrature LQ4
c
       elseif(nordin.eq.4) then
c
         ami1 = 0.3500212d00
           ami2 = dsqrt ( 1.d00 - 2.d00 * ami1 * ami1 )
c
         weight = 1.d00 / 3.d00
c
```

```
          ami(1) = ami1
          eta(1) = ami1
          zeta(1) = ami2
c
          w(1) = weight
c
          call rotoct (1)
c
c
          ami(2) = ami1
          eta(2) = ami2
          zeta(2) = ami1
c
          w(2) = weight
c
          call rotoct (2)
c
c
          ami(3) = ami2
          eta(3) = ami1
          zeta(3) = ami1
c
          w(3) = weight
c
          call rotoct (3)
c
c   Parameters for the quadrature LQ6
c
       elseif(nordin.eq.6) then
c
c
          ami1 = 0.2666355d00
c
          cost = 2.d00 * ( 1.d00 - 3.d00 * ami1 * ami1 )
     c                                 /  dfloat ( nordin - 2 )
            ami2 = dsqrt ( ami1 * ami1 + cost )
c
            ami3 = dsqrt ( ami1 * ami1 + 2.d00 * cost )
c
          weigh1 = 0.1761263d00
            weigh2 = ( 1.d00 - 3.d00 * weigh1 ) / 3.d00
c
c
          ami(1) = ami1
          eta(1) = ami1
          zeta(1) = ami3
c
          w(1) = weigh1
c
          call rotoct (1)
c
c
          ami(2) = ami1
          eta(2) = ami3
          zeta(2) = ami1
c
          w(2) = weigh1
c
          call rotoct (2)
c
c
          ami(3) = ami3
          eta(3) = ami1
          zeta(3) = ami1
c
          w(3) = weigh1
c
          call rotoct (3)
c
c
          ami(4) = ami2
          eta(4) = ami2
          zeta(4) = ami1
c
          w(4) = weigh2
c
          call rotoct (4)
c
c
          ami(5) = ami2
          eta(5) = ami1
          zeta(5) = ami2
c
          w(5) = weigh2
c
          call rotoct (5)
c
c
          ami(6) = ami1
          eta(6) = ami2
          zeta(6) = ami2
```

```
c
        w(6) = weigh2
c
        call rotoct (6)
c
c   Parameters for the quadrature LQ8
c
      elseif(nordin.eq.8) then
c
c
        ami1 = 0.2182179d00
c
        cost = 2.d00 * ( 1.d00 - 3.d00 * ami1 * ami1 )
     &                              /  dfloat ( nordin - 2 )
          ami2 = dsqrt ( ami1 * ami1 + cost )
c
          ami3 = dsqrt ( ami1 * ami1 + 2.d00 * cost )
c
          ami4 = dsqrt ( ami1 * ami1 + 3.d00 * cost )
c
        weigh1 = 0.1209877d00
          weigh2 = 0.0907407d00
          weigh3 = ( 1.d00 - 3.d00 * weigh1 - 6.d00 * weigh2 )
c
c
        ami(1) = ami1
        eta(1) = ami1
        zeta(1) = ami4
c
        w(1) = weigh1
c
        call rotoct (1)
c
c
        ami(2) = ami1
        eta(2) = ami4
        zeta(2) = ami1
c
        w(2) = weigh1
c
        call rotoct (2)
c
c
        ami(3) = ami4
        eta(3) = ami1
        zeta(3) = ami1
c
        w(3) = weigh1
c
        call rotoct (3)
c
c
        ami(4) = ami1
        eta(4) = ami2
        zeta(4) = ami3
c
        w(4) = weigh2
c
        call rotoct (4)
c
c
        ami(5) = ami2
        eta(5) = ami1
        zeta(5) = ami3
c
        w(5) = weigh2
c
        call rotoct (5)
c
c
        ami(6) = ami1
        eta(6) = ami3
        zeta(6) = ami2
c
        w(6) = weigh2
c
        call rotoct (6)
c
c
        ami(7) = ami2
        eta(7) = ami3
        zeta(7) = ami1
c
        w(7) = weigh2
c
        call rotoct (7)
c
c
        ami(8) = ami3
        eta(8) = ami1
        zeta(8) = ami2
```

```
c
         w(8) = weigh2
c
         call rotoct (8)
c
c
         ami(9) = ami3
         eta(9) = ami2
         zeta(9) = ami1
c
         w(9) = weigh2
c
         call rotoct (9)
c
c
         ami(10) = ami2
         eta(10) = ami2
         zeta(10) = ami2
c
         w(10) = weigh3
c
         call rotoct (10)
c
          endif
c
      do im = 1,nm
c
c   Assignement of the corresponding diretions by rotation around the coordinate axes
c
          do imr = 1,nm
            if ( (ami(imr).eq.-ami(im)).and.(eta(imr).eq.eta(im))
     &             .and.(zeta(imr).eq.zeta(im)) ) irx(im) = imr
            if ( (ami(imr).eq.ami(im)).and.(eta(imr).eq.-eta(im))
     &             .and.(zeta(imr).eq.zeta(im)) ) iry(im) = imr
            if ( (ami(imr).eq.ami(im)).and.(eta(imr).eq.eta(im))
     &             .and.(zeta(imr).eq.-zeta(im)) ) irz(im) = imr

           enddo
c
c   Check on the normalisation of direction cosines
c
      check = ami(im)*ami(im) + eta(im)*eta(im) + zeta(im)*zeta(im)
c
      write(7,100) im,ami(im),eta(im),zeta(im),w(im),check
c
      enddo
c
      return
  100 format(1x,i5,5(1x,e14.7))
      end
c-----------------------------------------------------------------------c
c                                                                       c
c    Subroutine to generate direction in the octants with   c
c    negative direction cosine                                    c
c                                                                       c
c-----------------------------------------------------------------------c
      subroutine rotoct (i)
      implicit double precision (a-h,o-z)
c
      include 'tras3dc.for'
c
      im = i + noct
c
c   ami < 0 , eta > 0 , zeta > 0
c
      w(im) = w(i)
c
      ami(im) = - eta(i)
        eta(im) = ami(i)
        zeta(im) = zeta(i)
c
c   ami < 0 , eta < 0 , zeta > 0
c
      im = i + noct * 2
c
      w(im) = w(i)
c
      ami(im) = - ami(i)
        eta(im) = - eta(i)
        zeta(im) = zeta(i)
c
c   ami > 0 , eta < 0 , zeta > 0
c
      im = i + noct * 3
c
      w(im) = w(i)
c
      ami(im) = eta(i)
        eta(im) = - ami(i)
        zeta(im) = zeta(i)
c
```

```
c   ami > 0 , eta > 0 , zeta < 0
c
      im = i + noct * 4
c
      w(im) = w(i)
c
      ami(im) = ami(i)
        eta(im) = eta(i)
        zeta(im) = - zeta(i)
c
c   ami < 0 , eta > 0 , zeta < 0
c
      im = i + noct * 5
c
      w(im) = w(i)
c
      ami(im) = - eta(i)
        eta(im) = ami(i)
        zeta(im) = - zeta(i)
c
c   ami < 0 , eta < 0 , zeta < 0
c
      im = i + noct * 6
c
      w(im) = w(i)
c
      ami(im) = - ami(i)
        eta(im) = - eta(i)
        zeta(im) = - zeta(i)
c
c   ami > 0 , eta < 0 , zeta < 0
c
      im = i + noct * 7
c
      w(im) = w(i)
c
      ami(im) = eta(i)
        eta(im) = - ami(i)
        zeta(im) = - zeta(i)
c
      return
        end
c-----------------------------------------------------------------------c
c                                                                       c
c    Subroutine for solving 3D transport equations                      c
c                                                                       c
c-----------------------------------------------------------------------c
      subroutine trasp
      implicit double precision (a-h,o-z)
c
      include 'tras3dc.for'
c
c   Zeroing the scalar flux
c
       do i = 1,nx
          do j = 1,ny
           do k = 1,nz
c
          phi(i,j,k) = 0.d00
c
          enddo
           enddo
          enddo
c
c  Loop on the directions
c
      do im = 1,nm
c
      write(*,100) im
      write(6,100) im
  100 format(1x,' Processamento della direzione ',i6)
c
c  Discussion of the sign of direction cosines and
c  boundary conditions: assignef flux or reflection
c
c
c  BC along the x axis
c
       if(ami(im).gt.0.d00) then
         i1 = 1
       i2 = nx
          ist = 1
c
        do j = 1,ny
           do k = 1,nz
              if(phiawf.ge.0.d00) then
            phiax(1,j,k) = phiawf
              else
            phiax(1,j,k) = phiawm(j,k,irx(im))
              endif
         enddo
```

```
          enddo
c
         else
         i1 = nx
       i2 = 1
         ist = - 1
c
        do j = 1,ny
           do k = 1,nz
              if(phiaef.ge.0.d00) then
           phiax(nx+1,j,k) = phiaef
              else
           phiax(nx+1,j,k) = phiaem(j,k,irx(im))
           endif
         enddo
          enddo
c
       endif
c
c  BC along the y axis
c

       if(eta(im).gt.0.d00) then
         j1 = 1
       j2 = ny
         jst = 1
c
        do i = 1,nx
           do k = 1,nz
              if(phiasf.ge.0.d00) then
           phiay(i,1,k) = phiasf
              else
           phiay(i,1,k) = phiasm(i,k,iry(im))
              endif
         enddo
          enddo
c
         else
         j1 = ny
       j2 = 1
         jst = - 1
c
        do i = 1,nx
           do k = 1,nz
              if(phianf.ge.0.d00) then
           phiay(i,ny+1,k) = phianf
              else
           phiay(i,ny+1,k) = phianm(i,k,iry(im))
              endif
         enddo
          enddo
c
       endif
c
c  BC along the z axis
c
       if(zeta(im).gt.0.d00) then
         k1 = 1
       k2 = nz
         kst = 1
c
        do i = 1,nx
           do j = 1,ny
              if(phiabf.ge.0.d00) then
           phiaz(i,j,1) = phiabf
              else
              phiaz(i,j,1) = phiabm(i,j,irz(im))
              endif
         enddo
          enddo
c
         else
         k1 = nz
       k2 = 1
         kst = - 1
c
        do i = 1,nx
           do j = 1,ny
              if(phiatf.ge.0.d00) then
           phiaz(i,j,nz+1) = phiatf
              else
           phiaz(i,j,nz+1) = phiatm(i,j,irz(im))
              endif
         enddo
          enddo
c
       endif
c
c  Solution of neuutron balance equations
c  sweeping the three spatial axes accordin to the sign of direction cosines
c
```

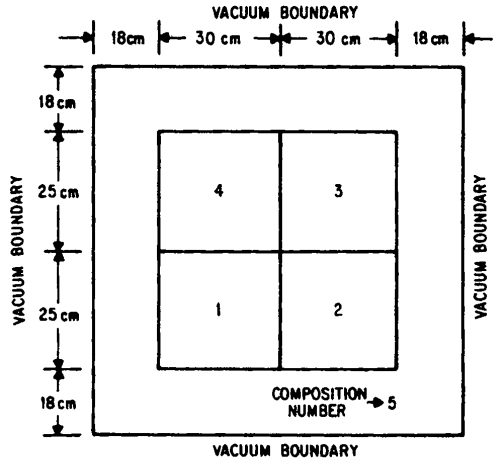
```
        do i = i1,i2,ist
        dx = x(i+1) - x(i)
          auxx = 2.d00 * dabs(ami(im)) / dx
c
         do j = j1,j2,jst
         dy = y(j+1) - y(j)
         auxy = 2.d00 * dabs(eta(im)) / dy
c
          do k = k1,k2,kst
          dz = z(k+1) - z(k)
          auxz = 2.d00 * dabs(zeta(im)) / dz
c
c  Construction of some useful quantities
c
          if(ist.eq.1) then
                addx = auxx * phiax(i,j,k)
            else
                addx = auxx * phiax(i+1,j,k)
            endif
c
          if(jst.eq.1) then
                addy = auxy * phiay(i,j,k)
            else
                addy = auxy * phiay(i,j+1,k)
            endif
c
          if(kst.eq.1) then
                addz = auxz * phiaz(i,j,k)
            else
                addz = auxz * phiaz(i,j,k+1)
            endif
c
c   Evaluation of tye central flus by the neutron balanve
c
          ifixx = 0
          ifixy = 0
          ifixz = 0
c
   20     denom = auxx + auxy + auxz + sigt(i,j,k)
c
          phiac(i,j,k) = ( addx + addy + addz + q(i,j,k) ) / denom
c
c   Application of the diamond rule and, in case of negative interface flux,
c   use of the "step" fix-up rule, then recalculating the central flux
c
          if(ist.eq.1) then
c
                  if(ifixx.eq.0) then
            phiax(i+1,j,k) = 2.d00 * phiac(i,j,k) - phiax(i,j,k)
              else
            phiax(i+1,j,k) = phiac(i,j,k)
              endif
c
                 if(phiax(i+1,j,k).lt.0.d00) then
                      auxx = dabs(ami(im)) / dx
                           addx = auxx * phiax(i,j,k)
                 ifixx = ifixx + 1
c                write(*,111) im,i,j,k
c                write(6,111) im,i,j,k
  111           format(1x,' Fix-up lungo x: direzione ',i5,' nodo ',3i6)
                goto 20
                endif
c
            else
c
                   if(ifixx.eq.0) then
              phiax(i,j,k) = 2.d00 * phiac(i,j,k) - phiax(i+1,j,k)
                else
              phiax(i,j,k) = phiac(i,j,k)
                endif
c
                 if(phiax(i,j,k).lt.0.d00) then
                      auxx = dabs(ami(im)) / dx
                           addx = auxx * phiax(i+1,j,k)
                 ifixx = ifixx + 1
c                write(*,111) im,i,j,k
c                write(6,111) im,i,j,k
                goto 20
                endif
c
            endif
c
          if(jst.eq.1) then
c
                   if(ifixy.eq.0) then
              phiay(i,j+1,k) = 2.d00 * phiac(i,j,k) - phiay(i,j,k)
                else
              phiay(i,j+1,k) = phiac(i,j,k)
                endif
c
                 if(phiay(i,j+1,k).lt.0.d00) then
```

```
                    auxy = dabs(eta(im)) / dy
                         addy = auxy * phiay(i,j,k)
                  ifixy = ifixy + 1
c                 write(*,112) im,i,j,k
c                 write(6,112) im,i,j,k
  112           format(1x,' Fix-up lungo y: direzione ',i5,' nodo ',3i6)
                  goto 20
                  endif
c
             else
c
                   if(ifixy.eq.0) then
             phiay(i,j,k) = 2.d00 * phiac(i,j,k) - phiay(i,j+1,k)
               else
             phiay(i,j,k) = phiac(i,j,k)
               endif
c
                  if(phiay(i,j,k).lt.0.d00) then
                       auxy = dabs(eta(im)) / dy
                            addy = auxy * phiay(i,j+1,k)
                  ifixy = ifixy + 1
c                 write(*,112) im,i,j,k
c                 write(6,112) im,i,j,k
                  goto 20
                  endif
c
             endif
c
           if(kst.eq.1) then
c
                   if(ifixz.eq.0) then
             phiaz(i,j,k+1) = 2.d00 * phiac(i,j,k) - phiaz(i,j,k)
               else
             phiaz(i,j,k+1) = phiac(i,j,k)
               endif
c
                  if(phiaz(i,j,k+1).lt.0.d00) then
                       auxz = dabs(zeta(im)) / dz
                            addz = auxz * phiaz(i,j,k)
                  ifixz = ifixz + 1
c                 write(*,113) im,i,j,k
c                 write(6,113) im,i,j,k
  113           format(1x,' Fix-up lungo z: direzione ',i5,' nodo ',3i6)
                  goto 20
                  endif
             else
c
                   if(ifixz.eq.0) then
             phiaz(i,j,k) = 2.d00 * phiac(i,j,k) - phiaz(i,j,k+1)
               else
             phiaz(i,j,k) = phiac(i,j,k)
               endif
c
                  if(phiaz(i,j,k).lt.0.d00) then
                       auxz = dabs(zeta(im)) / dz
                            addz = auxz * phiaz(i,j,k+1)
                  ifixz = ifixz + 1
c                 write(*,113) im,i,j,k
c                 write(6,113) im,i,j,k
                  goto 20
                  endif
c
             endif
c
c   Assignment of the contribution of the direction ot the scalar flux
c
          phi(i,j,k) = phi(i,j,k) + 0.125d00 * phiac(i,j,k) * w(im)
c
c  Assignement of the angular flux on thelateral surface
c  to allow imposing pure reflection boundary conditions
c
            if(i.eq.1) then
            phiawm(j,k,im) = phiax(i,j,k)
              elseif(i.eq.nx) then
                   phiaem(j,k,im) = phiax(nx+1,j,k)
                   endif
c
            if(j.eq.1) then
            phiasm(i,k,im) = phiay(i,j,k)
              elseif(j.eq.ny) then
                   phianm(i,k,im) = phiay(i,ny+1,k)
                   endif
c
            if(k.eq.1) then
            phiabm(i,j,im) = phiaz(i,j,k)
              elseif(k.eq.nz) then
                   phiatm(i,j,im) = phiaz(i,j,nz+1)
                   endif
c
          enddo
c
```

```
        enddo
c
       enddo
c
      enddo
c
      return
        end
c-----------------------------------------------------------------------c
c                                                                       c
c    Subroutine for undating the emission density                       c
c    from the new scattering source                                     c
c                                                                       c
c-----------------------------------------------------------------------c
      subroutine sscat
      implicit double precision (a-h,o-z)
c
      include 'tras3dc.for'
c
c  Assignement of density emission
c
        do i = 1,nx
           do j = 1,ny
            do k = 1,nz
             q(i,j,k) = sour(i,j,k) + sigs(i,j,k) * phi(i,j,k)
c
            enddo
           enddo
          enddo
c
      return
        end
```

Suggested personal work: new calculations cases to be analysed:

a) Benchmark Problem IAEA EIR-2 (v. H. Khalil, *Nucl. Sci. Eng.*, 90, pp.263-280, 1985)



COMPOSITION	Σ_t (cm^{-1})	Σ_s (cm^{-1})	SOURCE STRENGTH ($cm^{-3}\cdot s^{-1}$)
1	0.60	0.53	1.0
2	0.48	0.20	0.0
3	0.70	0.66	1.0
4	0.65	0.50	0.0
5	0.90	0.89	0.0

Fig. 1. Geometry and material properties for the IAEA EIR-2 benchmark problem.

TABLE III

Comparison of SYNAPSE and TWODANT Results for the IAEA EIR-2 Problem*

	SYNAPSE	TWODANT	TWODANT (Reference)
Number of X-Y mesh cells, $I \times J$	36 × 30	54 × 46	144 × 120
Number of synthetic iterations	5	10	6
Solution time[a] (s)	13	~31	~127
Average flux by composition[b] ($cm^{-2}\cdot s^{-1}$)			
1	1.1956+1[c]	1.1958+1	1.1960+1
2	5.4050−1	5.4038−1	5.3968−1
3	1.9193+1	1.9200+1	1.9202+1
4	8.3562−1	8.3358−1	8.3364−1
5	1.5285+0	1.5291+0	1.5263+0

*Directional approximation = S_8.
[a]Solution times do not include the "overhead" associated with input processing and output editing.
[b]See Fig. 1.
[c]Read as 1.1956×10^1.

b) Prism with imposed inner flux
c) Different sources in a prisamtic domain
d) Parametric analyses of the already analysed cases, discussing the results
e) Comparison between the results that can be obtained by different approximations for the angular discretisation