DE LA RECHERCHE À L'INDUSTRIE

HIGHER ORDER ACCELERATED MOC METHOD

by
Laurent Graziano & Simone Santandrea

www.cea.fr

METHOD OF CHARACTERISTICS (MOC)

- *source iterative solution*

  (one-group problem)

  $$\downarrow$$

  $$\psi(\mathbf{r},\Omega)$$

$$\left.\begin{aligned}(\Omega\cdot\nabla+\Sigma)\psi^{(n+1)} &= q^{(n)} \\ {\psi_{in}}^{(n+1)} &= \beta{\psi_{out}}^{(n)}+\psi_0\end{aligned}\right\}$$

$$q^{(n)} = H\psi^{(n)} + S$$

$$albedo\ \ operator\ \ \beta : \psi_{out} \rightarrow \psi_{in}$$

- *angular approximation*

$$S_N = \{w_n, \Omega_n, n = 1, N\} \rightarrow \frac{1}{4\pi}\int_{(4\pi)} d\Omega f(\Omega) \sim \sum_n w_n f(\Omega_n)$$

- *flat flux approximation over homogeneous regions*

$$D = \underset{i}{\cup} D_i \ ,\ D_i \ \ of\ \ homogeneous\ \ support$$

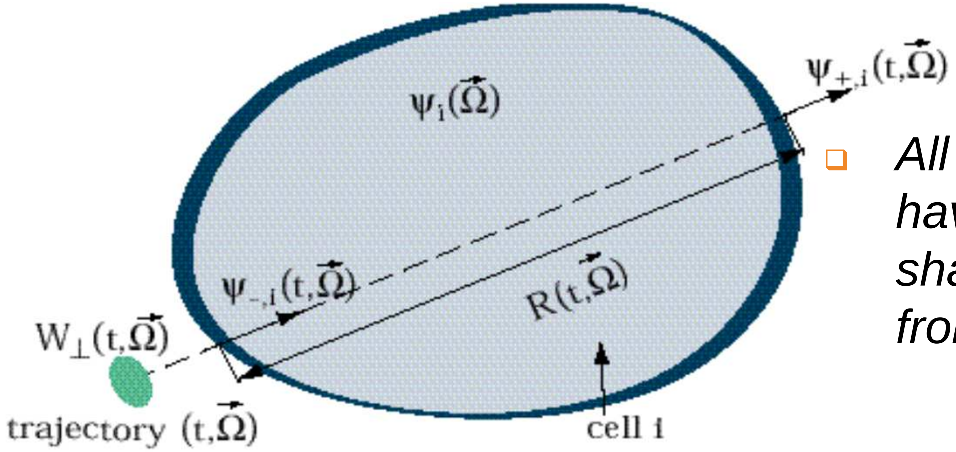
$$\psi(\mathbf{r},\Omega) \sim \sum_i \psi_i(\Omega)\theta_i(\mathbf{r})$$

MOC 1 : DISCRETIZATION SCHEME

- *numerical implementation based on trajectories*

  (in 2D XY 'planar' trajectories are lifted to polar directions)

$\Omega \in S_N$:



- *All following pictures have been shamelessly taken from R. Sanchez*

- *balance equation along a trajectory*

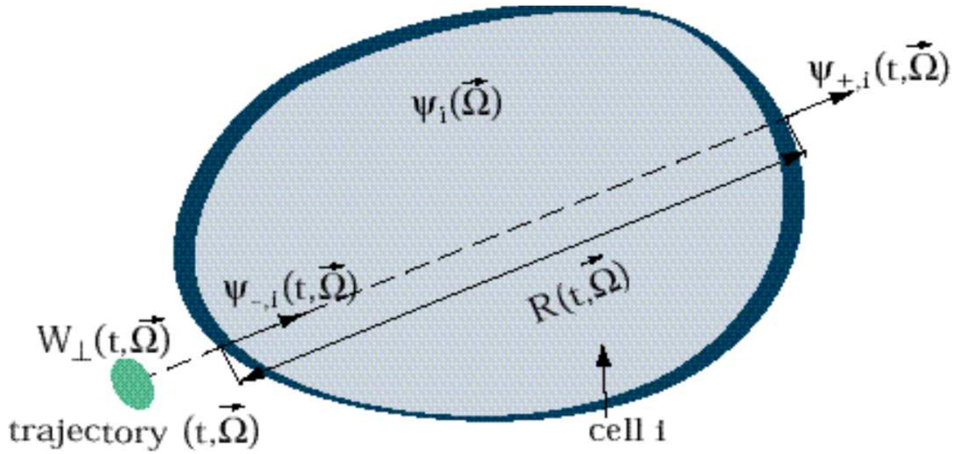
$$\psi_{+,i}(t,\Omega) - \psi_{-,i}(t,\Omega) + \Sigma_i R_i(t,\Omega)\psi_i(t,\Omega) = R_i(t,\Omega)q_i^{(n)}(\Omega)$$

$$\downarrow$$

$$V_i\psi_i(\Omega) = \int_{(i)} dr\,\psi(r,\Omega) \sim \sum_{t\|\Omega} \omega_\perp(t,\Omega)R_i(t,\Omega)\psi_i(t,\Omega)$$

MOC 2 : DISCRETIZATION SCHEME

$\mathbf{\Omega} \in S_N$:



- *propagation equation across a region* *(flat source approximation)*

$$\underbrace{\psi_+(\mathbf{r},\mathbf{\Omega}) = e^{-\tau(r,r_{in})}\psi_-(\mathbf{r}_{in},\mathbf{\Omega}) + \int_0^{R(t,\mathbf{\Omega})} dR' e^{-\tau(\mathbf{r},\mathbf{r}')} q(\mathbf{r}',\mathbf{\Omega})}_{\downarrow} \qquad (\mathbf{r}' = \mathbf{r} - R'\mathbf{\Omega})$$

$$\psi_{+,i}(t,\mathbf{\Omega}) = T_i(t,\mathbf{\Omega}) \times \psi_{-,i}(t,\mathbf{\Omega}) + E_i(t,\mathbf{\Omega}) \times q_i^{(n)}(\mathbf{\Omega})$$

transmission & escape coefficients : $T_i(t,\mathbf{\Omega}) = e^{-\Sigma_i R_i(t,\mathbf{\Omega})}, \quad E_i(t,\mathbf{\Omega}) = \dfrac{1 - T_i(t,\mathbf{\Omega})}{\Sigma_i}$

HIGHER ORDER BALANCE EQUATION: A CONSISTENCY PROBLEM

Defining a scalar product on a chord

$$\langle f, g \rangle_L = \int_0^L dt\, f(t) g(t),$$

the following generalized average balance per chord can be written

$$\Sigma_r \left\langle \vec{P}, \Psi_r \right\rangle_L = \left\langle \vec{P}, \vec{P} \right\rangle_L \vec{q}_r + \vec{P}(0)\Psi_r(0) - \vec{P}(L)\Psi_r(L) + \left\langle \frac{\partial \vec{P}}{\partial t}, \Psi_r \right\rangle_L,$$ (Sanchez 2012)

so that defining the polynomial coupling region matrix

$$\bar{\bar{P P}}(\vec{\Omega}) = \frac{1}{V_r} \int_r d\vec{r}\, \vec{P}(\tilde{z}) \otimes \vec{P}(\tilde{z}) \simeq \frac{1}{V_r(\vec{\Omega})} \sum_{\substack{t \| \vec{\Omega} \\ t \cap r}} \left\langle \vec{P}, \vec{P} \right\rangle_{L_t},$$

the « polynomial angular » balance equation is:

$$\boxed{\Sigma_r\, '\vec{\Psi}_r(\vec{\Omega}) = \bar{\bar{P P}}(\vec{\Omega}) \cdot \vec{q}_r(\vec{\Omega}) - \Delta \vec{J}_r(\vec{\Omega}) + \mu\, \bar{\bar{C}}\, '\vec{\Psi}_r(\vec{\Omega}).}$$

$$\bar{\bar{C}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{\Delta z/2} & 0 & 0 & 0 \\ 0 & \frac{2}{\Delta z/2} & 0 & 0 \\ 0 & 0 & \frac{3}{\Delta z/2} & 0 \\ 0 & 0 & \ddots & \ddots \end{bmatrix}.$$

BALANCE FROM ANGLE TO MOMENTS

The collision/fission operators use angular moments in the place of angular fluxes:

$$\vec{\Phi}_r^n = \oint \frac{d\vec{\Omega}}{4\pi} A_n(\vec{\Omega})\vec{\Psi}_r(\vec{\Omega}).$$

Recall also that a « correspondence » exists between spatial moment and coefficients

$$\vec{\Psi}_r(\vec{\Omega}) = P\bar{\bar{P}}^{-1} \cdot \ '\vec{\Psi}_r(\vec{\Omega})$$

If you define then the suite of angular-polynomial function base

$$\vec{\mathcal{Z}}(\tilde{z},\vec{\Omega}) = \{A^0(\vec{\Omega})P_0(\tilde{z}), A^1(\vec{\Omega})P_0(\tilde{z}), ..., A^0(\vec{\Omega})P_1(\tilde{z}), A^1(\vec{\Omega})P_1(\tilde{z}), ...\}$$

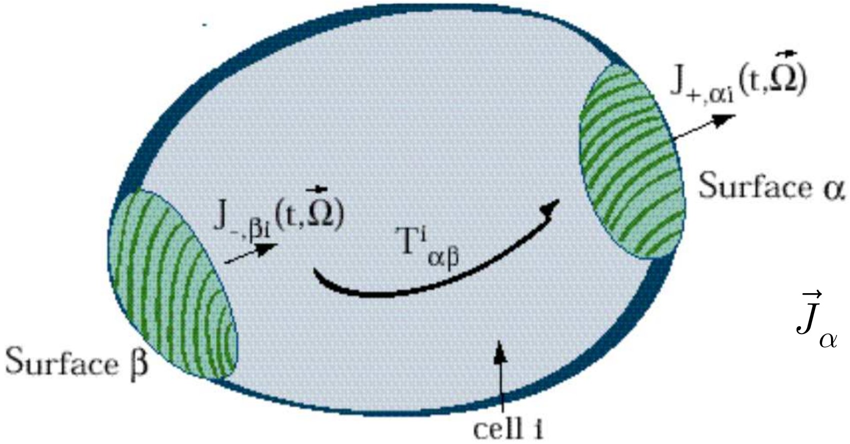
and project with over this in angle-space you get

$$\Sigma_r \ '\vec{\Phi}_r = \mathcal{Z}\mathcal{Z} \cdot \vec{q}_r - \oint \frac{d\vec{\Omega}}{4\pi} \vec{A}(\vec{\Omega}) \otimes \Delta\vec{J}_r(\vec{\Omega}) + \mathcal{D} \cdot {'\vec{\Phi}_r}. \quad \mathcal{Z}\mathcal{Z} = \oint \frac{d\vec{\Omega}}{4\pi} \left(\vec{A}(\vec{\Omega}) \otimes \vec{A}(\vec{\Omega})\right) \otimes P\bar{\bar{P}}(\vec{\Omega})$$

which in the typical “free” iteration scheme (index “i”) takes an easy to solve lower triangular form:

$$\Sigma_r \ '\vec{\Phi}_{r,p}^{\,i} = \left(\mathcal{Z}\mathcal{Z} \cdot \vec{q}_r^{\,i-1}\right)_p - \frac{1}{4\pi} \oint d\vec{\Omega} \ \vec{A}(\vec{\Omega}) \otimes \left(\Delta\vec{J}_r(\vec{\Omega})\right)_p^i + \frac{p}{\Delta z/2} \cdot \bar{\bar{\alpha}}_p \cdot \ '\vec{\Phi}_{r,p-1}^{\,i}$$

DP_N : SYNTHETIC ACCELERATION



$$\psi_{\pm}(\mathbf{r},\Omega) \sim \vec{A}_S(\Omega) \cdot \sum_{\alpha \in \partial i} \vec{\psi}_{\pm,\alpha} \theta_\alpha(\mathbf{r})$$

$$\vec{J}_\alpha = \frac{1}{4\pi S_\alpha} \int_\alpha dS \int_{(4\pi)} d\Omega |\Omega n| \vec{Z}\, \psi \underset{DPn}{\sim} A_{\alpha,+} \vec{\psi}_{\alpha,+}$$

$$A_{\alpha,+} = \frac{1}{4\pi S_\alpha} \int_\alpha dS \int_{(2\pi+)} d\Omega \vec{Z} \otimes \vec{Z} = ppA_{\alpha,-}$$

- *propagation & balance equations :*

$$\begin{bmatrix} \vec{J}^+_{\alpha_v} \\ \vec{J}^+_{\alpha_h} \end{bmatrix} = \sum_{\beta \in r} \begin{bmatrix} \mathcal{T}_{\alpha_v^+ \beta_v^-} & \mathcal{T}_{\alpha_v^+ \beta_h^-} \\ \mathcal{T}_{\alpha_h^+ \beta_v^-} & \mathcal{T}_{\alpha_h^+ \beta_h^-} \end{bmatrix} \cdot \begin{bmatrix} \vec{\Phi}^-_{\beta_v} \\ \vec{\Phi}^-_{\beta_h} \end{bmatrix} + \begin{bmatrix} \mathcal{E}_{\alpha_v^+} \\ \mathcal{E}_{\alpha_h^+} \end{bmatrix} \cdot \vec{q}_r,$$

$$(\Sigma_r - \mathcal{D}) \cdot{}' \vec{\Phi}_r = \mathcal{Z}\mathcal{Z}_D \cdot \vec{q}_r - \frac{1}{V_r} \sum_{\alpha \in r} \left(\vec{J}^+_\alpha - \vec{J}^-_\alpha \right),$$

- *After "some" algebra a multi-collisional version of the DP_N operator is used to solve:*

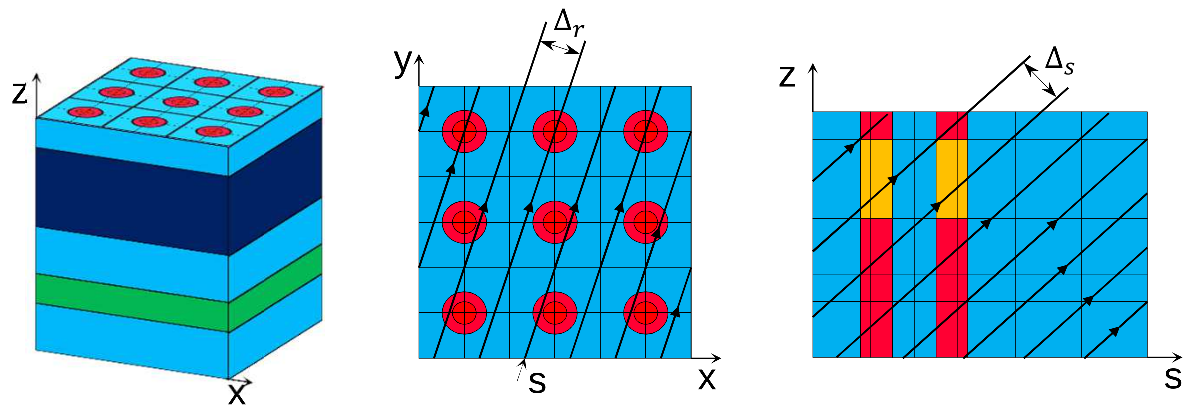
$$\vec{J}^+ = {}^\dagger\mathcal{T} \cdot \vec{J}^- + {}^\dagger\mathcal{E} \cdot \vec{q}_r^{\,ext},$$

$${}^\dagger\mathcal{T} = \left(\tilde{\mathcal{T}} + \vec{\mathcal{E}}^+ \Sigma^g_{s,r} \cdot {}^\dagger\mathcal{I} \right),$$

$${}^\dagger\mathcal{E} = \vec{\mathcal{E}}^+ \left(\mathcal{I}_d + \Sigma^g_{r,s} \cdot {}^\dagger\mathcal{C} \right)$$

GOING TO AXIAL 3D GEOMETRIES

The basic difficulty for 3D MOC calculation is that we cannot store realistic 3D tracking data. To avoid this we consider only (at the beginning!) 3D axial geometries:



For these geometries the 3D tracking can be decomposed into 2 phases:

1. Tracking a general 2D geometry on the x-y plane
2. Tracking a cartesian geometry on the s-z plane

Only 1 need to be stored but reconstruct 2 can be too expensive!

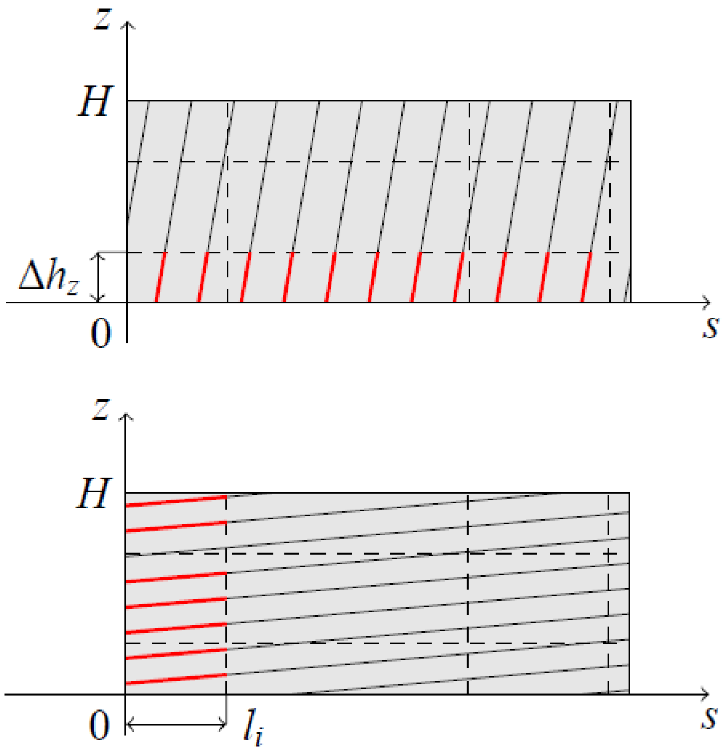
OPTIMIZED 3D TRACKING

Thanks to axial regularity the set of 3D chords can be decomposed into a low number of classes that not only allow to reduce memory but also to decrease computational cost.

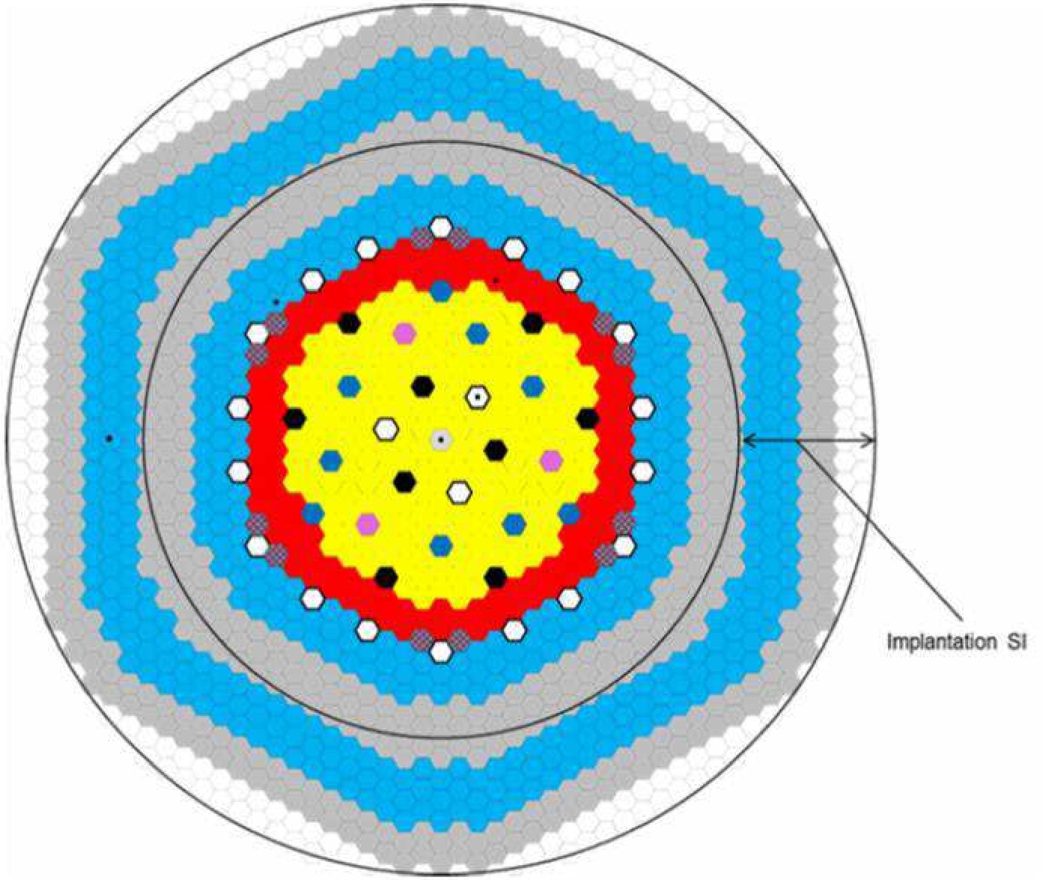
Thus, transmission coefficients,

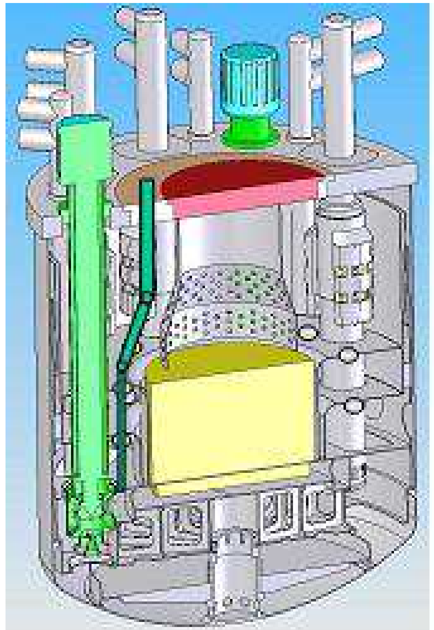
$$T_i(t,\mathbf{\Omega}) = e^{-\Sigma_i R_i(t,\mathbf{\Omega})}$$

are computed only per class and medium.



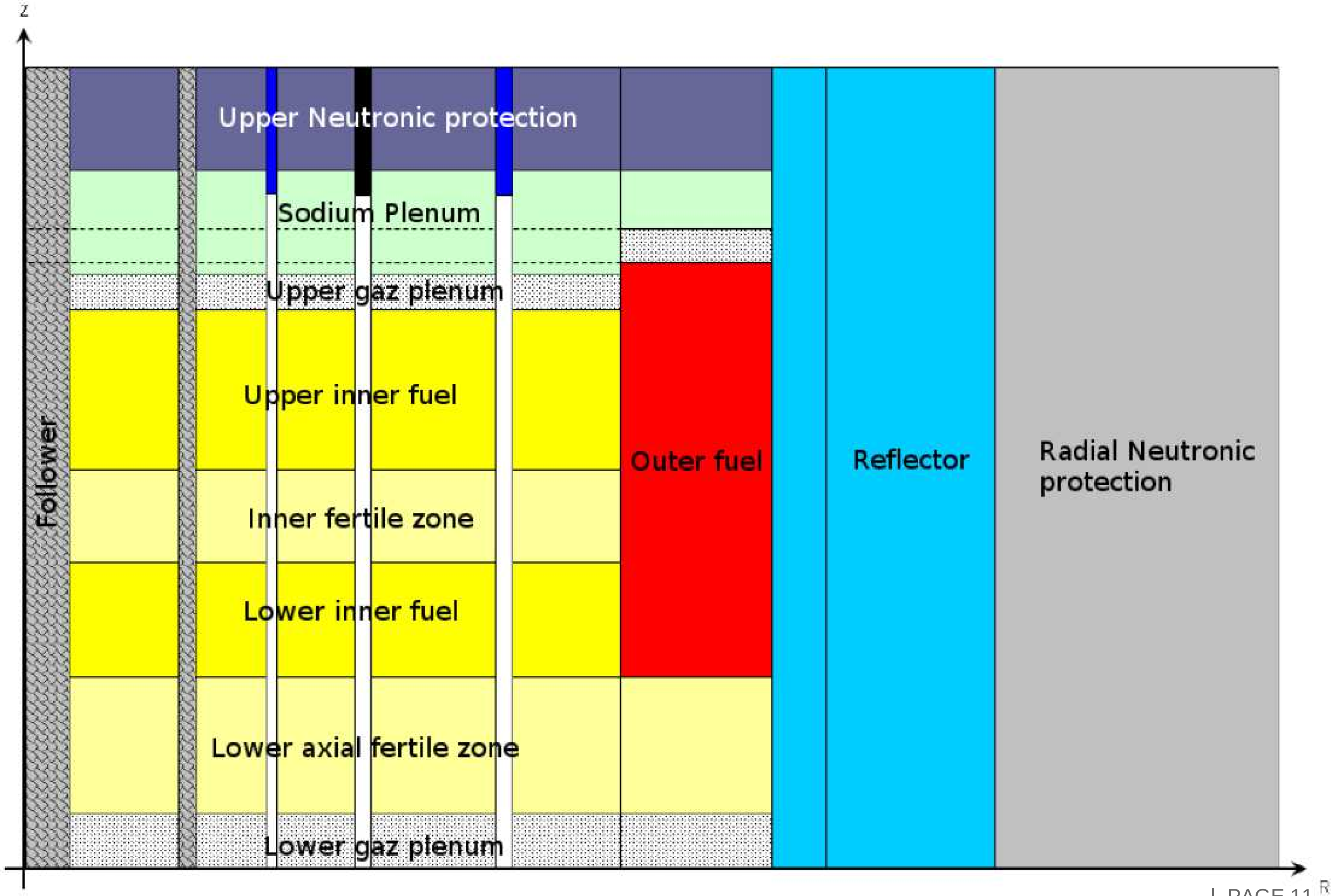
ASTRID REACTOR EXAMPLE: 2D SECTION

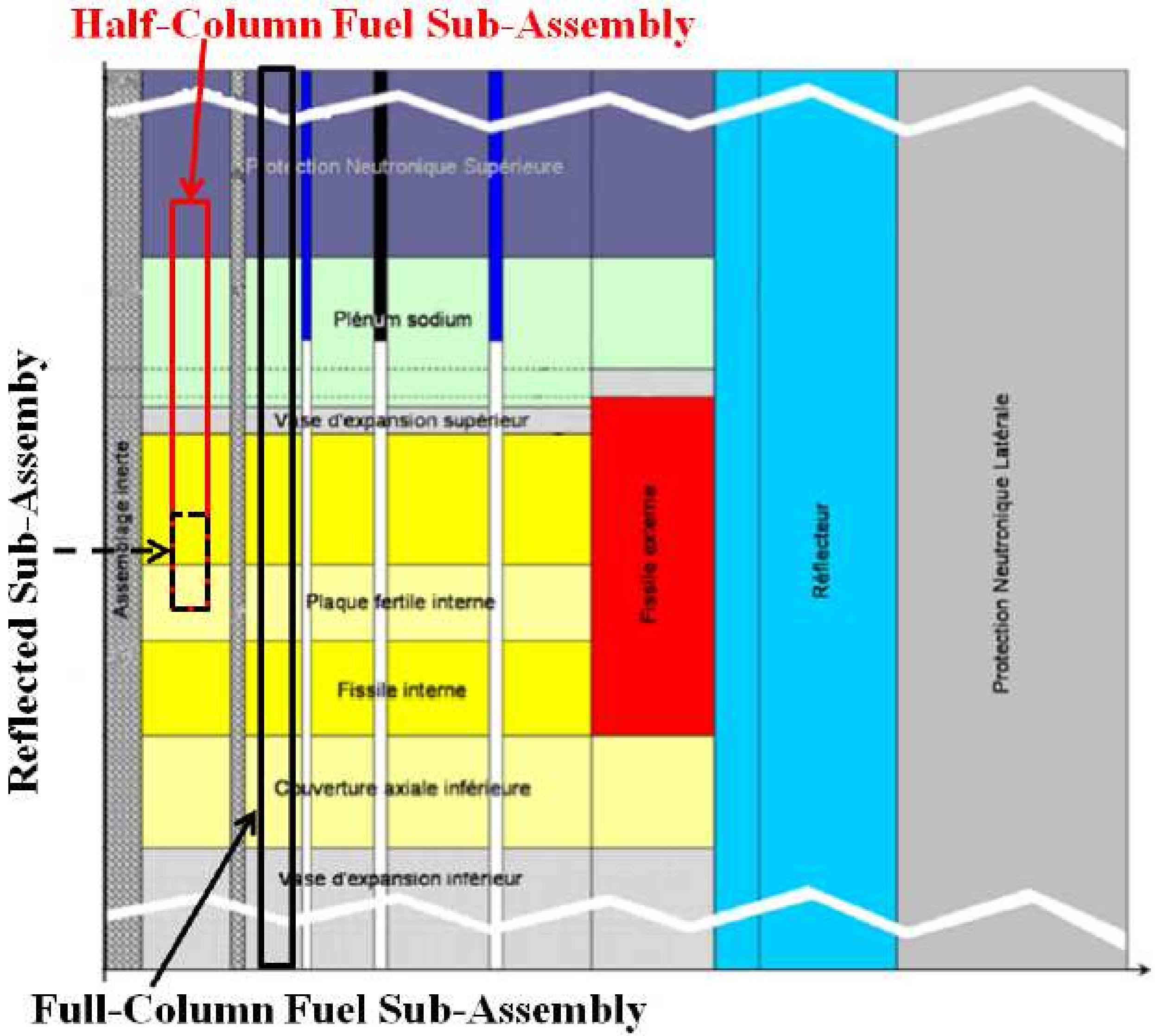




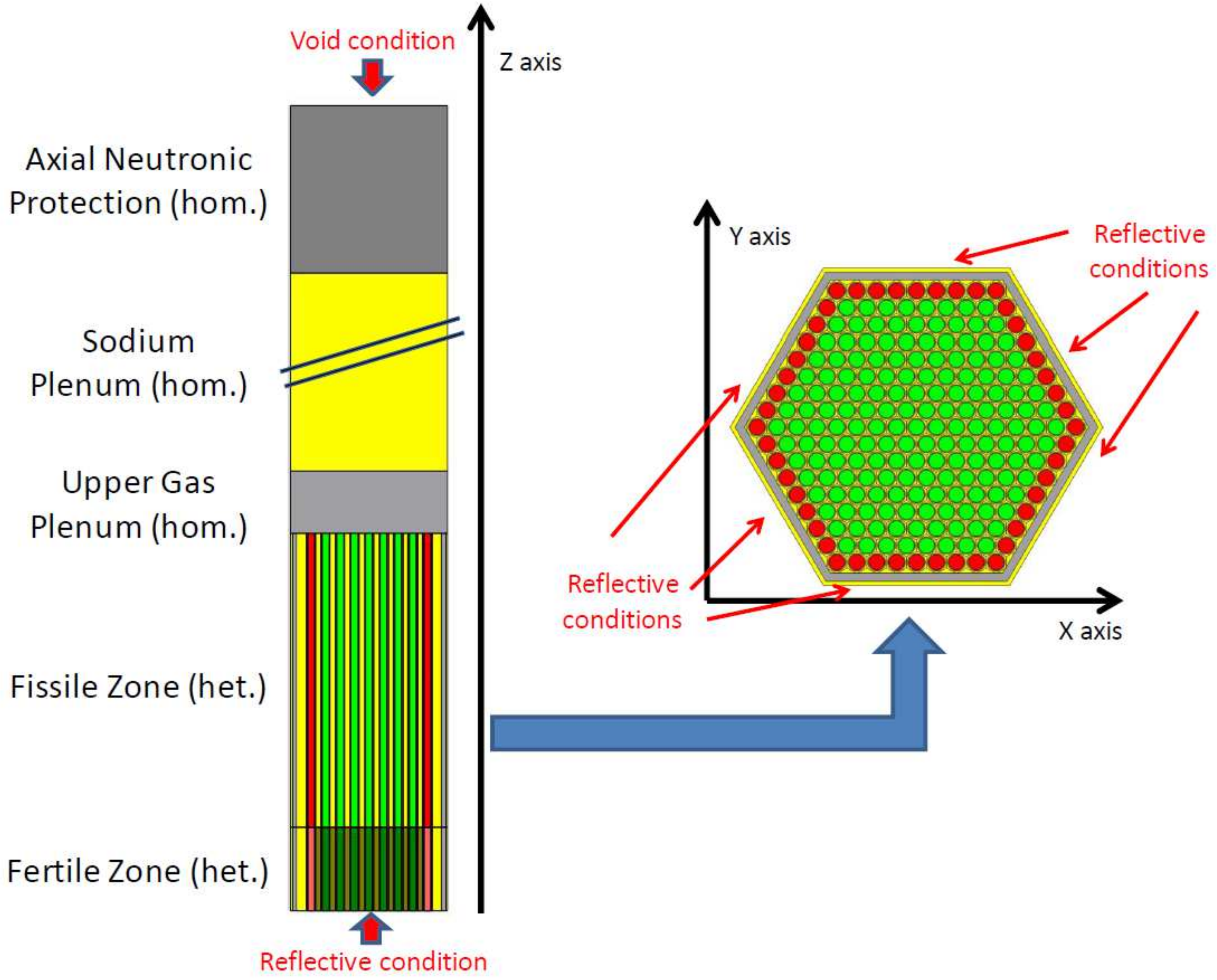
Assemblage inerte (position pouvant être occupée par un assemblage absorbant)
Assemblages combustibles cœur interne
Assemblages combustibles cœur externe
Assemblages RBC
Assemblages RBD
DIMEP B
Assemblages Réflecteurs
Assemblages PNL
Assemblages (DCS-P)-H
Assemblages (DCS-M)-TT
Positions pouvant être utilisées
Positions Réflecteurs pouvant être occupées par des assemblages combustibles

ASTRID REACTOR: AXIAL VIEW

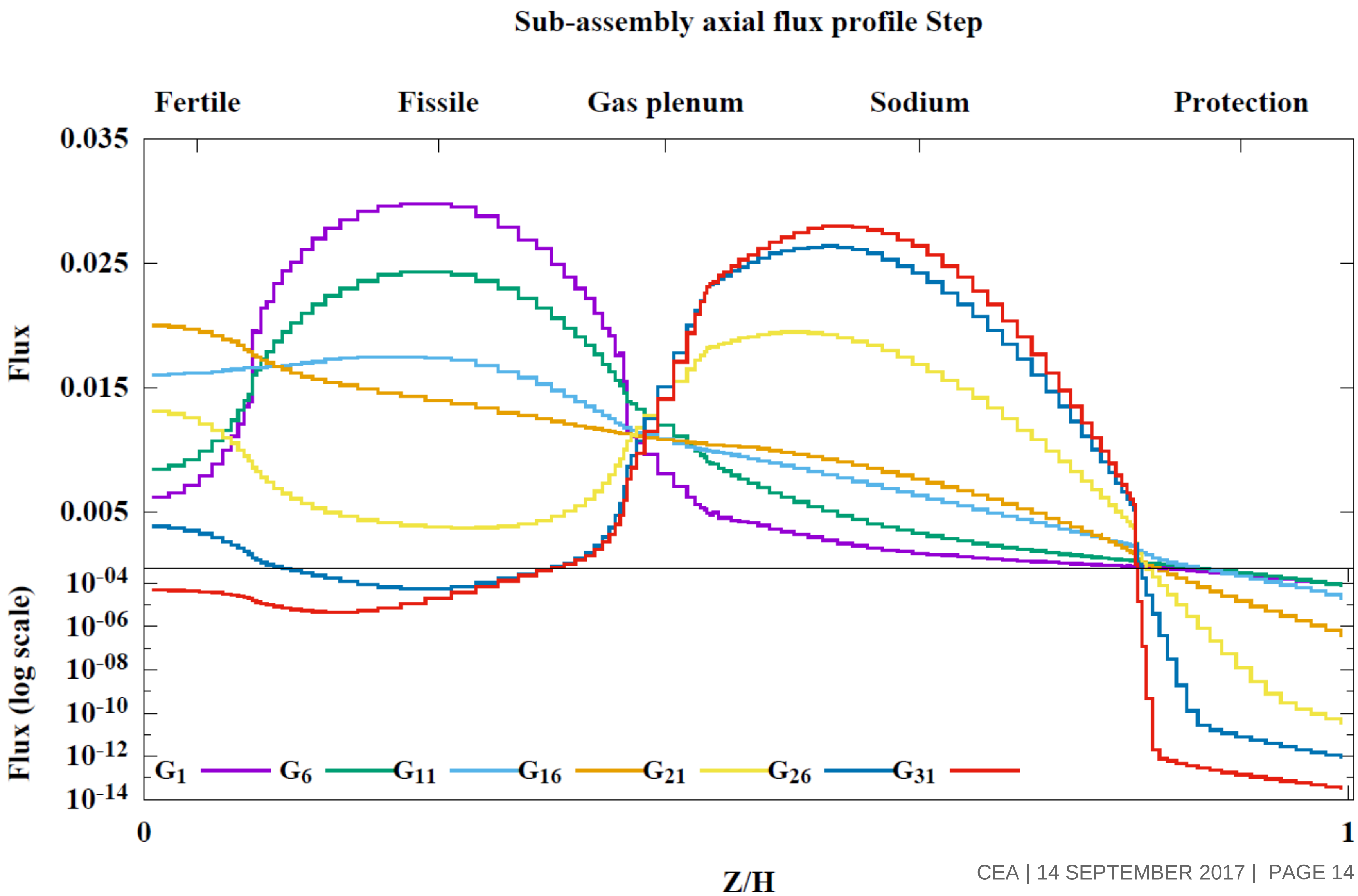



Half-Column Fuel Sub-Assembly
Plénum sodium
Vase d'expansion supérieur
Assemblage inerte
Plaque fertile interne
Fissile interne
Fissile externe
Réflecteur
Protection Neutronique Latérale
Couverture axiale inférieure
Vase d'expansion inférieur
Reflected Sub-Assemby
Full-Column Fuel Sub-Assembly

ASTRID REACTOR: SECOND CALCULATION



ASTRID REACTOR: SECOND CALCULATION



POLYNOMIAL BASIS DEFINITION

Polynomial basis to express fluxes and sources moments:

$$\vec{P}(\tilde{z}_r) = \{\tilde{z}_r^p = \left(\frac{z_r - \bar{z}_r}{\Delta z_r/2}\right)^p, \quad 0 \leq p \leq N_p\} \qquad \tilde{z}_r \in [-1, 1]$$

POLYNOMIAL BASIS DEFINITION

Polynomial basis to express fluxes and sources moments:

$$\vec{P}(\tilde{z}_r) = \{\tilde{z}_r^p = \left(\frac{z_r - \bar{z}_r}{\Delta z_r/2}\right)^p, \quad 0 \leq p \leq N_p\} \qquad \tilde{z}_r \in [-1, 1]$$

$$q(\vec{r}, \vec{\Omega}) = \sum_{n=1}^{N_m} A_n(\vec{\Omega}) \cdot q^n(\vec{r})$$

POLYNOMIAL BASIS DEFINITION

Polynomial basis to express fluxes and sources moments:

$$\vec{P}(\tilde{z}_r) = \{\tilde{z}_r^p = \left(\frac{z_r - \bar{z}_r}{\Delta z_r/2}\right)^p, \quad 0 \le p \le N_p\} \qquad \tilde{z}_r \in [-1, 1]$$

$$q(\vec{r}, \vec{\Omega}) = \sum_{n=1}^{N_m} A_n(\vec{\Omega}) \cdot q^n(\vec{r})$$

Step approximation $\longrightarrow$ $q^n(\vec{r}) \simeq q_r^n$

POLYNOMIAL BASIS DEFINITION

Polynomial basis to express fluxes and sources moments:

$$\vec{P}(\tilde{z}_r) = \{\tilde{z}_r^p = \left(\frac{z_r - \bar{z}_r}{\Delta z_r/2}\right)^p, \quad 0 \le p \le N_p\} \qquad \tilde{z}_r \in [-1, 1]$$

$$q(\vec{r}, \vec{\Omega}) = \sum_{n=1}^{N_m} A_n(\vec{\Omega}) \cdot q^n(\vec{r})$$

Step approximation $\longrightarrow$ $q^n(\vec{r}) \simeq q_r^n$

Polynomial approximation $\longrightarrow$ $\boxed{q^n(\vec{r}) = \sum_p^{N_p} P_p(\tilde{z}_r) \cdot q_{r,pol,p}^n}$

POLYNOMIAL BASIS DEFINITION

Polynomial basis to express fluxes and sources moments:

$$\vec{P}(\tilde{z}_r) = \{\tilde{z}_r^p = \left(\frac{z_r - \bar{z}_r}{\Delta z_r/2}\right)^p, \quad 0 \leq p \leq N_p\} \qquad \tilde{z}_r \in [-1, 1]$$

$$q(\vec{r}, \vec{\Omega}) = \sum_{n=1}^{N_m} A_n(\vec{\Omega}) \cdot q^n(\vec{r})$$

Step approximation $\longrightarrow$ $q^n(\vec{r}) \simeq q_r^n$

Polynomial approximation $\longrightarrow$ $\boxed{q^n(\vec{r}) = \sum_{p}^{N_p} P_p(\tilde{z}_r) \cdot q_{r,pol,p}^n}$

$$\boxed{q(\vec{r}, \vec{\Omega}) = \vec{P}(\tilde{z}_r) \cdot \vec{q}_{r,pol}(\vec{\Omega})}$$

$$q_{r,pol,p}(\vec{\Omega}) = \sum_{n}^{N_m} A_n(\vec{\Omega}) \cdot q_{r,pol,p}^n$$

POLYNOMIAL TRANSMISSION EQUATION

- Polynomial transmission equation:

$$q(\vec{r},\vec{\Omega}) = \vec{P}(\tilde{z}_r)\cdot\vec{q}_{r,pol}(\vec{\Omega})$$

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \int_{t^{in}}^{t^{out}} dt'\; q\left(\vec{r}(t'),\vec{\Omega}\right)\; e^{-\Sigma_r(t^{out}-t')}$$

POLYNOMIAL TRANSMISSION EQUATION

- Polynomial transmission equation:

$$q(\vec{r},\vec{\Omega}) = \vec{P}(\tilde{z}_r)\cdot\vec{q}_{r,pol}(\vec{\Omega})$$

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \int_{t^{in}}^{t^{out}} dt' \; q\left(\vec{r}(t'),\vec{\Omega}\right)\; e^{-\Sigma_r (t^{out}-t')}$$

- Numerical polynomial transmission equation:

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} +$$
$$+\sum_{k=0}^{N_p} P_k(z^{in})\cdot\sum_{p=k}^{N_p} c_{pk}\;\mu^{p-k}\left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau)\frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}$$

POLYNOMIAL TRANSMISSION EQUATION

- Polynomial transmission equation:

$$q(\vec{r},\vec{\Omega}) = \vec{P}(\tilde{z}_r)\cdot\vec{q}_{r,pol}(\vec{\Omega})$$

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \int_{t^{in}}^{t^{out}} dt'\; q\left(\vec{r}(t'),\vec{\Omega}\right)\; e^{-\Sigma_r(t^{out}-t')}$$

- Numerical polynomial transmission equation:

Axial coordinate at the trajectory entering point in the r region

Binomial coefficient

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \\ + \sum_{k=0}^{N_p} P_k(z^{in}) \sum_{p=k}^{N_p} c_{pk}\,\mu^{p-k}\left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau)\frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}$$

POLYNOMIAL TRANSMISSION EQUATION

- Polynomial transmission equation:

$$q(\vec{r},\vec{\Omega}) = \vec{P}(\tilde{z}_r) \cdot \vec{q}_{r,pol}(\vec{\Omega})$$

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \int_{t^{in}}^{t^{out}} dt' \; q\left(\vec{r}(t'),\vec{\Omega}\right) \; e^{-\Sigma_r (t^{out}-t')}$$

- Numerical polynomial transmission equation:

Geometrical coefficients

Source term

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + $$
$$+ \sum_{k=0}^{N_p} P_k(z^{in}) \cdot \sum_{p=k}^{N_p} c_{pk}\, \mu^{p-k} \left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau) \frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}$$

POLYNOMIAL TRANSMISSION EQUATION

- Polynomial transmission equation:

$$q(\vec{r},\vec{\Omega}) = \vec{P}(\tilde{z}_r)\cdot\vec{q}_{r,pol}(\vec{\Omega})$$

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \int_{t^{in}}^{t^{out}} dt'\ q\left(\vec{r}(t'),\vec{\Omega}\right)\ e^{-\Sigma_r(t^{out}-t')}$$

- Numerical polynomial transmission equation:

Escape coefficient:

$$E_{p-k}(\tau) = \frac{1}{\Sigma_r^{(p-k)}}\int_{\tau(t^{in})}^{\tau(t^{out})} d\tau'\tau'^{p-k}e^{(\tau'-\tau(t^{in}))}$$

$$where \quad \tau = \Sigma_r t$$

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} +$$

$$+\sum_{k=0}^{N_p} P_k(z^{in})\cdot\sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k}\left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau)\frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}$$

STEP *VS* POLYNOMIAL TRANSMISSION

- Step transmission:

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

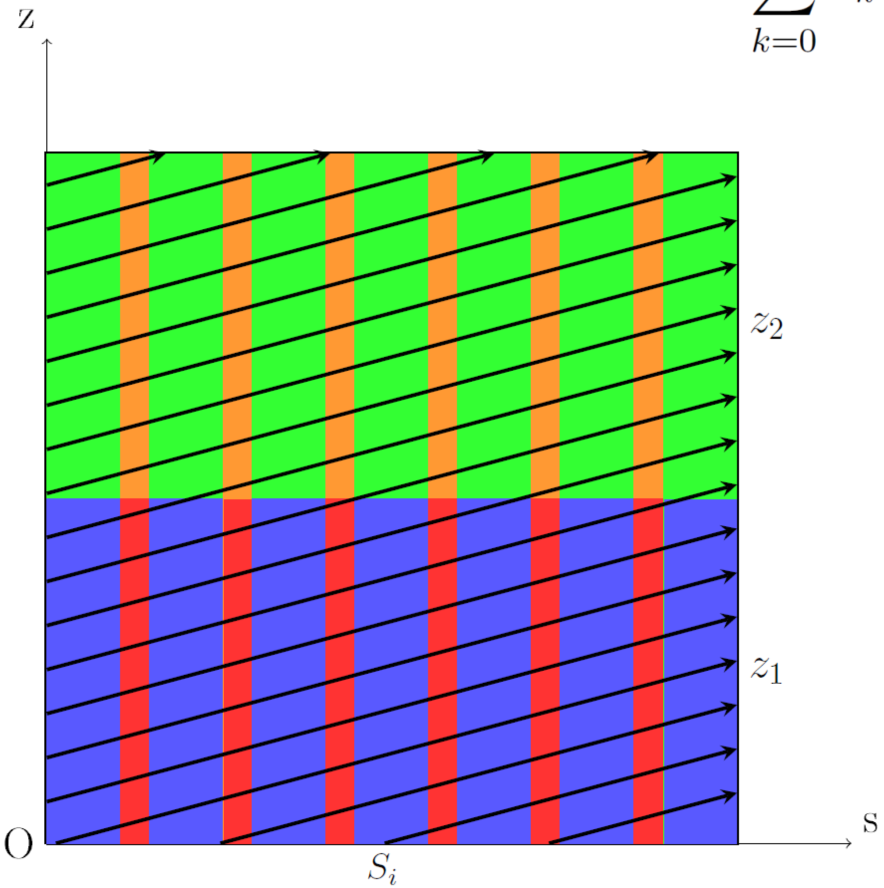
- Polynomial transmission:

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} +$$

$$+ \sum_{k=0}^{N_p} P_k(z^{in}) \cdot \sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k} \left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau) \frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}$$

CHORDS CLASSIFICATION

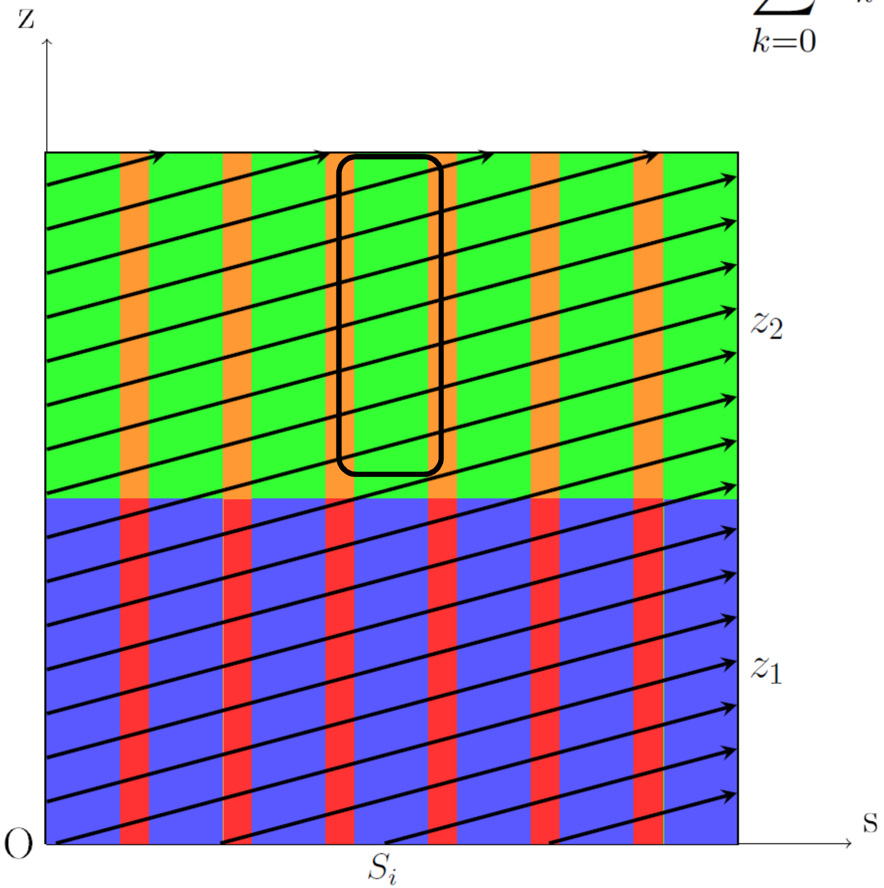
$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} +$$
$$+ \sum_{k=0}^{N_p} P_k(z^{in}) \cdot \sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k} \left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau) \frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}$$



CHORDS CLASSIFICATION

Thanks to chords classification…

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} +$$
$$+ \sum_{k=0}^{N_p} P_k(z^{in}) \cdot \sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k} \left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau) \frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}$$

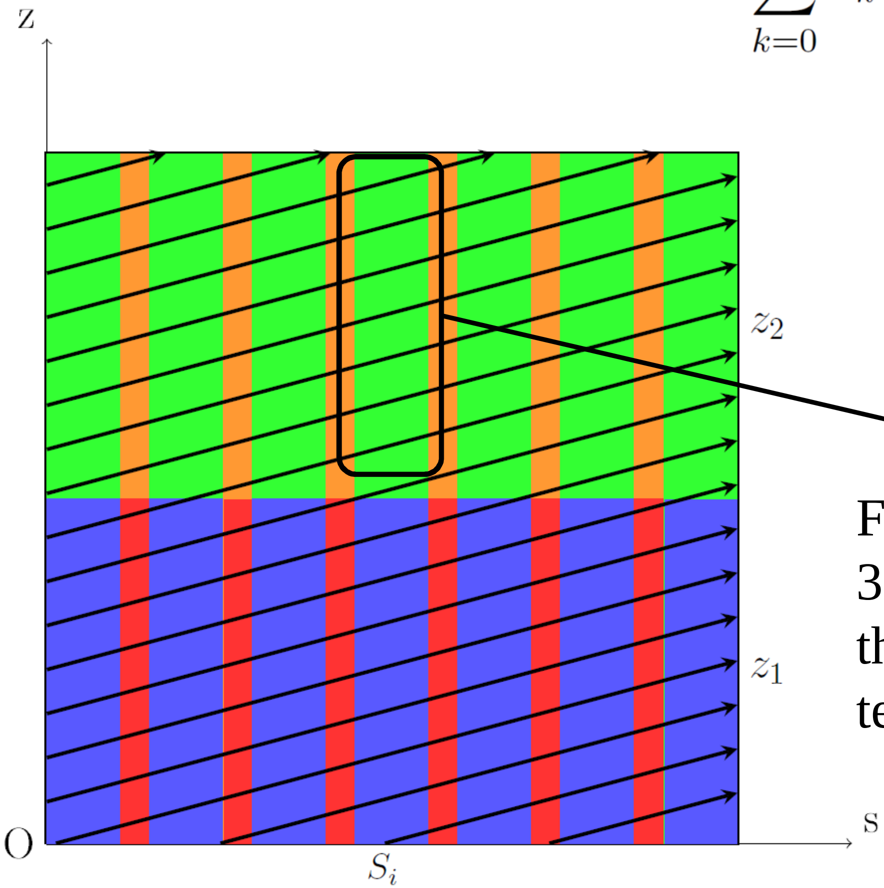


CHORDS CLASSIFICATION

Thanks to chords classification…

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} +$$

$$+ \sum_{k=0}^{N_p} P_k(z^{in}) \cdot \underbrace{\sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k} \left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau) \frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}}$$



For a given angle, z-plane and 2D-chord, each 3D chords with the same length, belongs to the same class and has the same values of this term

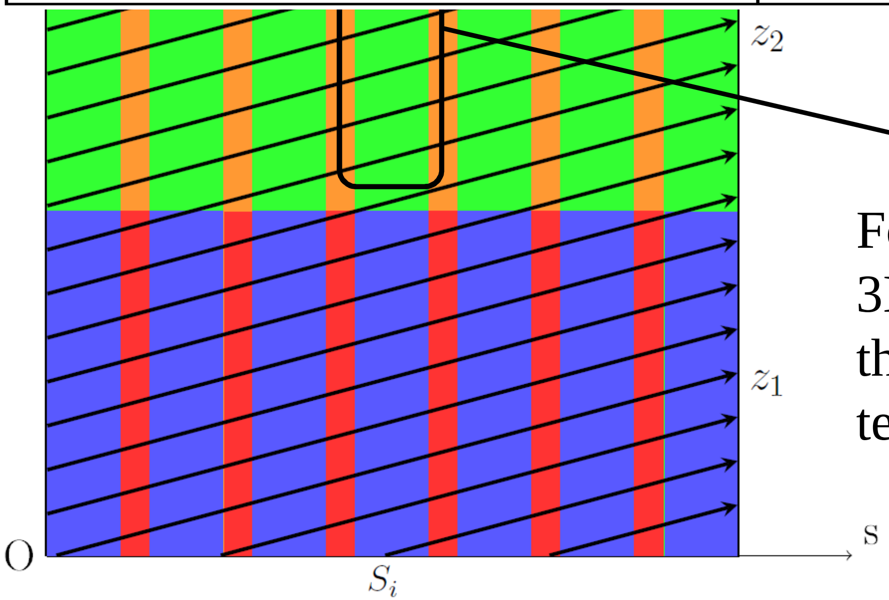
CHORDS CLASSIFICATION

Thanks to chords classification…

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} +$$

$$+ \sum_{k=0}^{N_p} P_k(z^{in}) \cdot \underbrace{\sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k} \left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau) \frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}}$$

• **Total number of chords**	10.39 M
• **Number of classes**	0.814 M



For a given angle, z-plane and 2D-chord, each 3D chords with the same length, belongs to the same class and has the same values of this term

CHORDS CLASSIFICATION

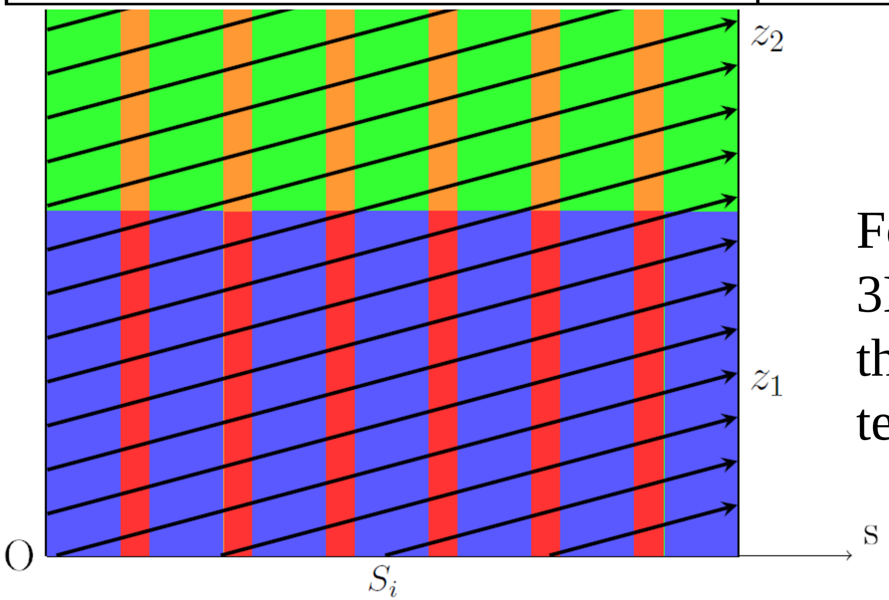
Thanks to chords classification…

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \sum_{k=0}^{N_p} P_k(z^{in})\cdot \underbrace{\sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k}\left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau)\frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}}$$

This is only computed for the ~ **8 %** of chords

z

• **Total number of chords**	10.39 M
• **Number of classes**	0.814 M



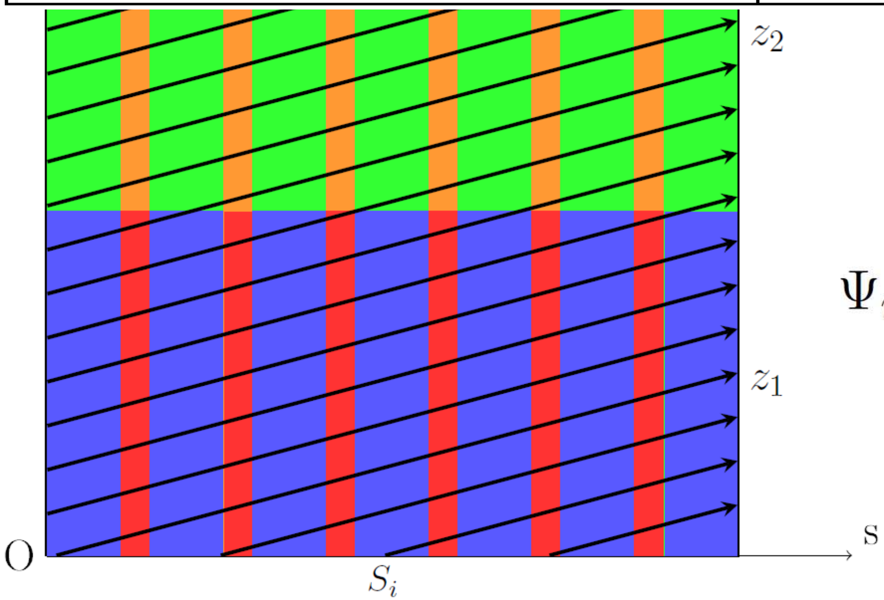
For a given angle, z-plane and 2D-chord, each 3D chords with the same length, belongs to the same class and has the same values of this term

Thanks to chords classification…

This is only computed for the ~ **8 %** of chords

$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \\ + \sum_{k=0}^{N_p} P_k(z^{in}) \cdot \underbrace{\sum_{p=k}^{N_p} c_{pk}\ \mu^{p-k} \left(\frac{2}{\Delta z}\right)^{p-k} E_{p-k}(\tau)\frac{\left(\vec{q}_{r,pol}(\vec{\Omega})\right)_p}{\Sigma_r}}$$

• **Total number of chords**	10.39 M
• **Number of classes**	0.814 M



$$\Psi_r(t^{out},\vec{\Omega}) = \Psi_r(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \vec{P}(z^{in})\cdot\vec{T}$$

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \vec{P}(z^{in}) \cdot \vec{T}$$

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \vec{P}(z^{in}) \cdot \vec{T}$$

1 floating point operation

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \vec{P}(z^{in}) \cdot \vec{T}$$

1 floating point operation

N_p floating point operations

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial

$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \vec{P}(z^{in}) \cdot \vec{T}$$

N_p floating point operations

1 floating point operation

Plus the information needed for the **balance equation**:

Step

$$\Psi_r(\vec{\Omega}) = \frac{1}{\Sigma_r}\left[q_r(\vec{\Omega}) - \frac{S_\perp}{V_r}\sum_{\substack{t\|\vec{\Omega}\\ t\cap r}}\left(\Psi(t^{out},\vec{\Omega}) - \Psi(t^{in},\vec{\Omega})\right)\right]$$

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step
$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial
$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \vec{P}(z^{in}) \cdot \vec{T}$$

$\boxed{N_p \text{ floating point operations}}$

1 floating point operation

Plus the information needed for the **balance equation**:

Step
$$\Psi_r(\vec{\Omega}) = \frac{1}{\Sigma_r}\left[q_r(\vec{\Omega}) - \frac{S_\perp}{V_r}\boxed{\sum_{\substack{t\|\vec{\Omega}\\ t\cap r}}\left(\Psi(t^{out},\vec{\Omega}) - \Psi(t^{in},\vec{\Omega})\right)}\right]$$

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step
$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right)\cdot\frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial
$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \vec{P}(z^{in})\cdot\vec{T}$$

$\boxed{\text{N}_\text{p}\text{ floating point operations}}$

1 floating point operation

Plus the information needed for the **balance equation**:

Step
$$\Psi_r(\vec{\Omega}) = \frac{1}{\Sigma_r}\left[q_r(\vec{\Omega}) - \frac{S_\perp}{V_r}\boxed{\sum_{\substack{t\|\vec{\Omega}\\ t\cap r}}\left(\Psi(t^{out},\vec{\Omega}) - \Psi(t^{in},\vec{\Omega})\right)}\right]$$

Polynomial
$$\tilde{\delta}_{r,p}(\vec{\Omega}) = \sum_{\substack{t\|\vec{\Omega}\\ t\cap r}}\left[P_p(z^{out})\cdot\Psi(t^{out}) - P_p(z^{in})\cdot\Psi(t^{in})\right]$$

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step
$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \left(1 - e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial
$$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega}) \cdot e^{-\Sigma_r l} + \vec{P}(\vec{z}^{in}) \cdot \vec{T}$$

$\boxed{\text{N}_\text{p} \text{ floating point operations}}$

1 floating point operation

Plus the information needed for the **balance equation**:

Step
$$\Psi_r(\vec{\Omega}) = \frac{1}{\Sigma_r}\left[q_r(\vec{\Omega}) - \frac{S_\perp}{V_r} \boxed{\sum_{\substack{t\|\vec{\Omega} \\ t\cap r}} \left(\Psi(t^{out},\vec{\Omega}) - \Psi(t^{in},\vec{\Omega})\right)} \right]$$

$\boxed{\text{N}_\text{p} \text{ floating point operations}}$

Polynomial
$$\tilde{\delta}_{r,p}(\vec{\Omega}) = \sum_{\substack{t\|\vec{\Omega} \\ t\cap r}} \left[P_p(\vec{z}^{out}) \cdot \Psi(t^{out}) - P_p(\vec{z}^{in}) \cdot \Psi(t^{in}) \right]$$

STEP *VS* POLYNOMIAL

- For a fair comparison:

Step $$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \left(1-e^{-\Sigma_r l}\right) \cdot \frac{q_r(\vec{\Omega})}{\Sigma_r}$$

Polynomial $$\Psi(t^{out},\vec{\Omega}) = \Psi(t^{in},\vec{\Omega})\cdot e^{-\Sigma_r l} + \vec{P}(z^{in}) \cdot \vec{T}$$

1 floating point operation

N_p floating point operations

Plus the information needed for the **balance equation**:

S $$\ldots ut,\vec{\Omega}) - \Psi(t^{in},\vec{\Omega})\Big)\Big]$$

They can be vectorized!

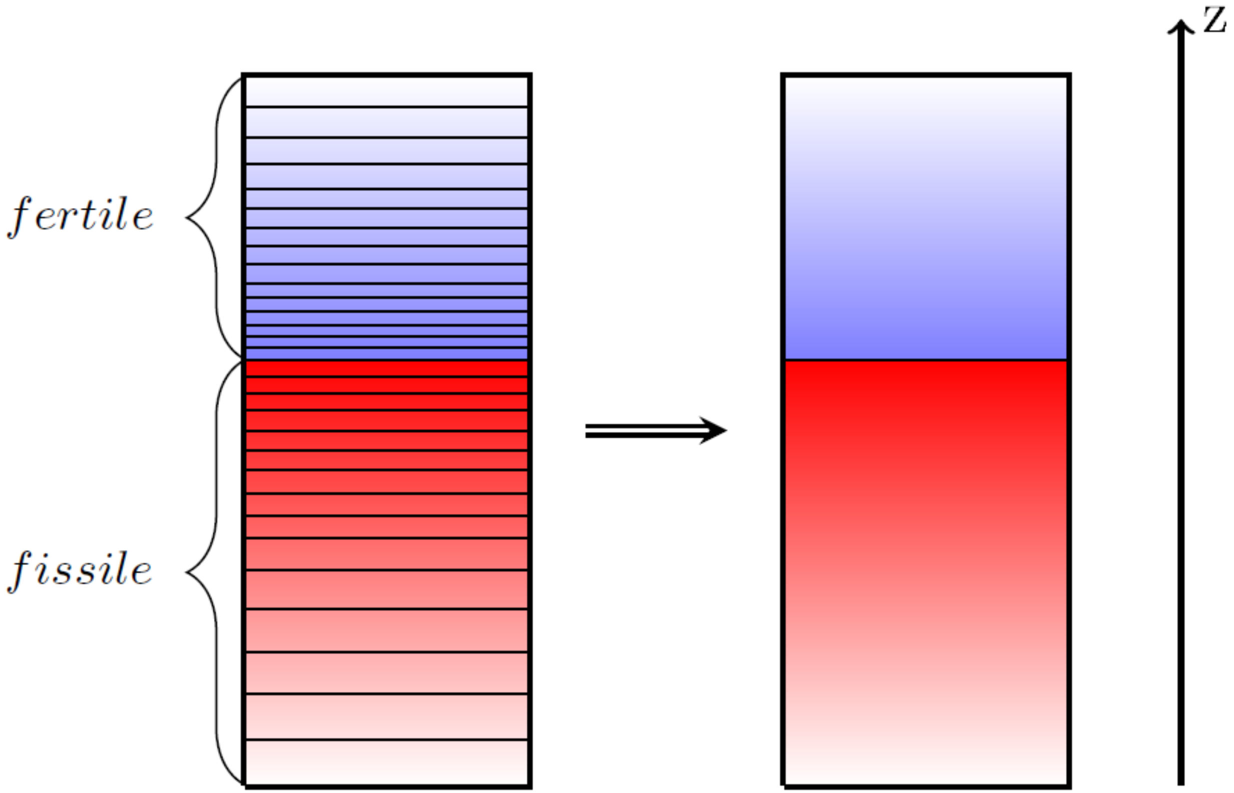
N_p floating point operations

P $$\ldots ut) \cdot \Psi(t^{out}) - P_p(z^{in})\cdot\Psi(t^{in})\big]$$

$t||\Omega$
$t\cap r$

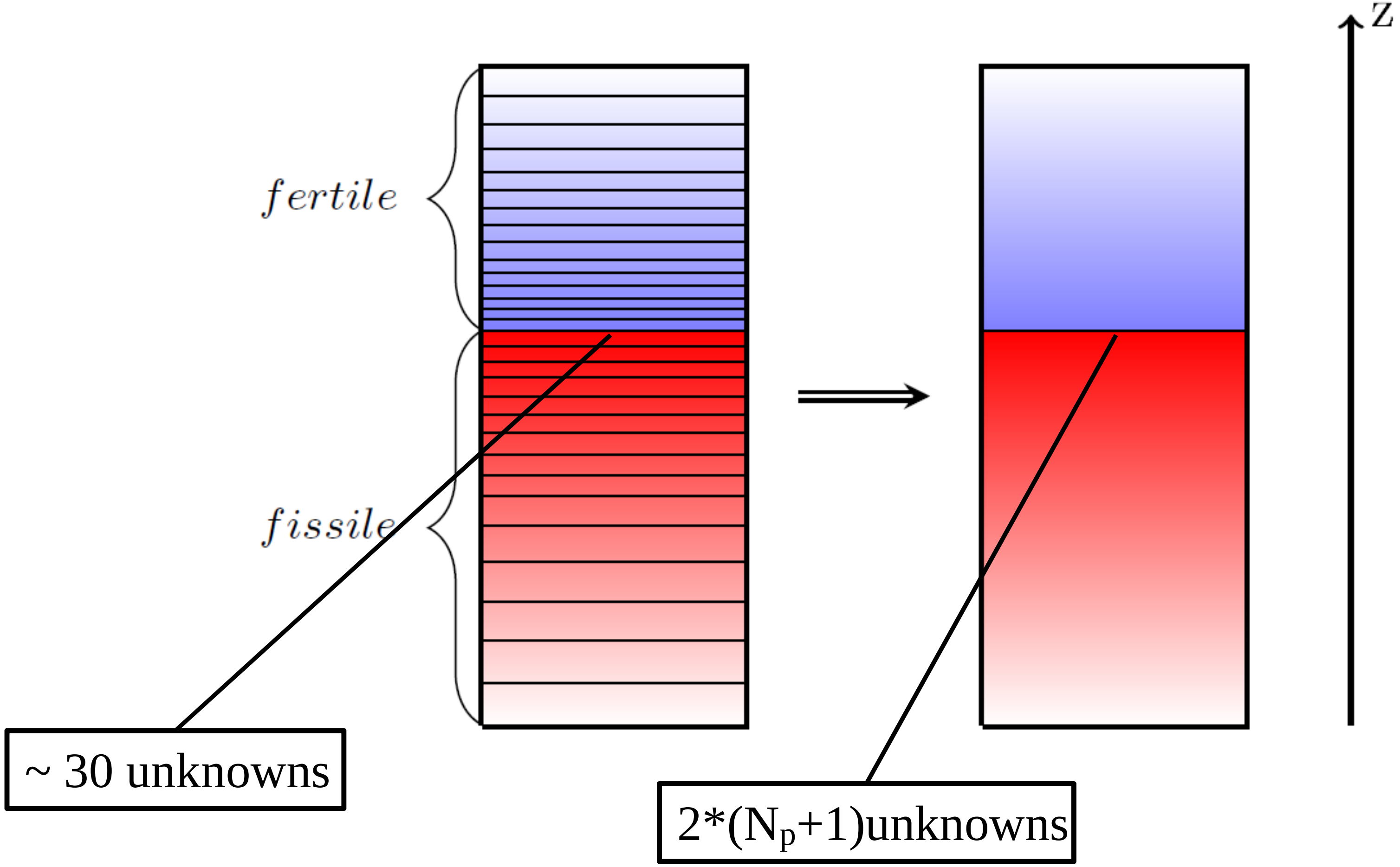
AXIAL DISCRETIZATION

Difference between the axial discretization needed in the Step Constant and in the Polynomial case:



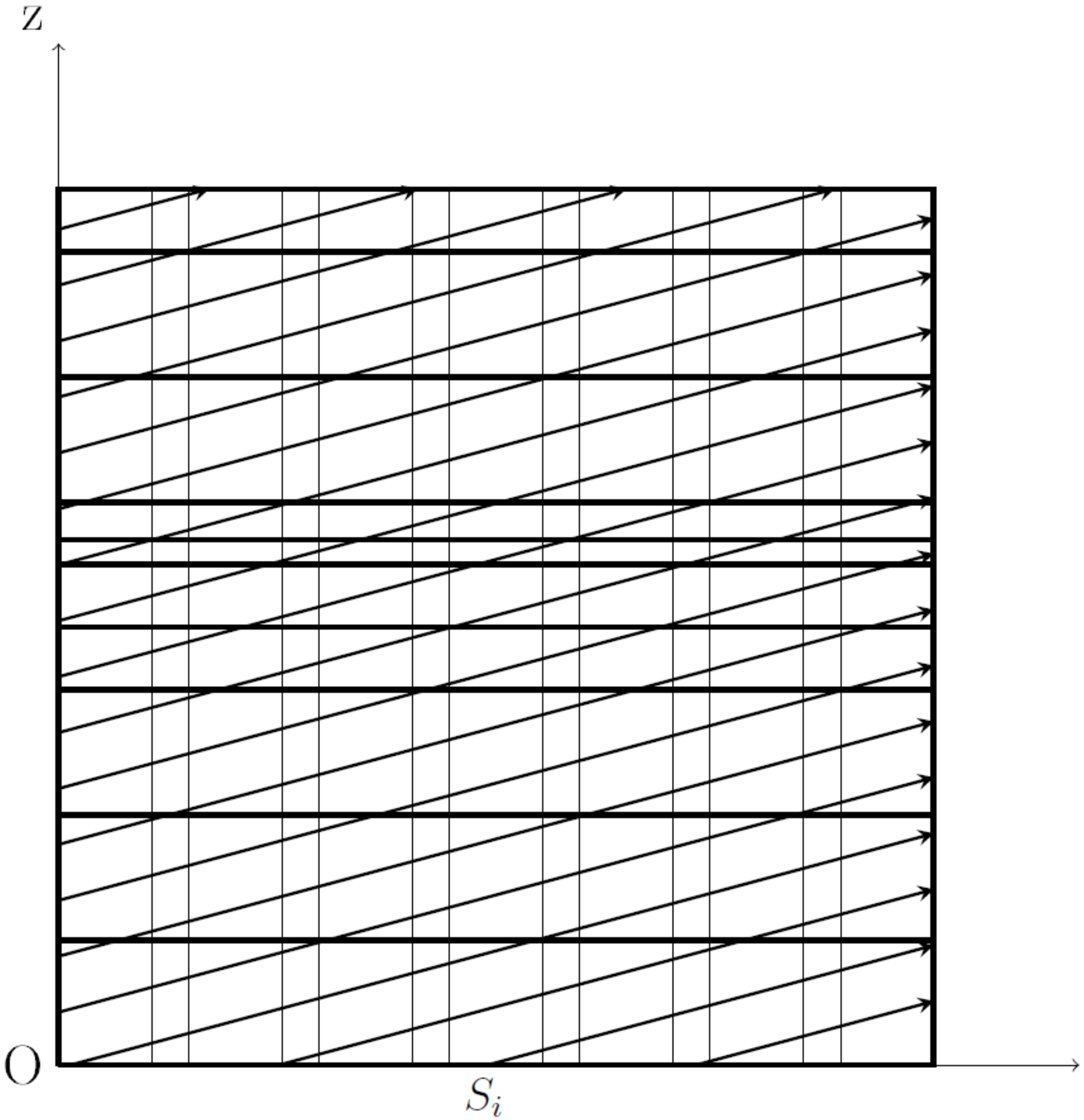
AXIAL DISCRETIZATION

Difference between the axial discretization needed in the Step Constant and in the Polynomial case:



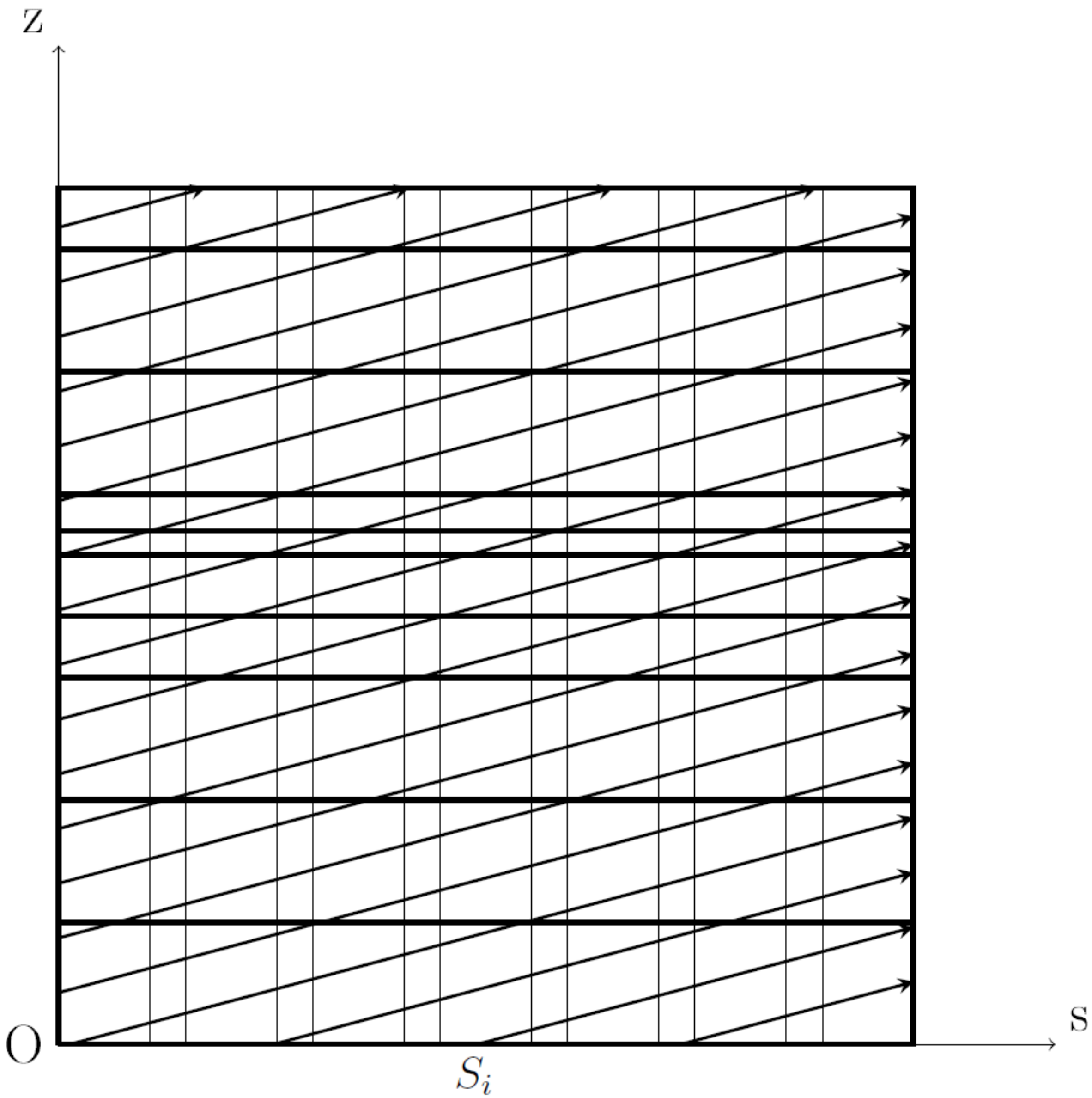
FINAL CONSIDERATIONS

- Higher numbers of floating point operations per chord



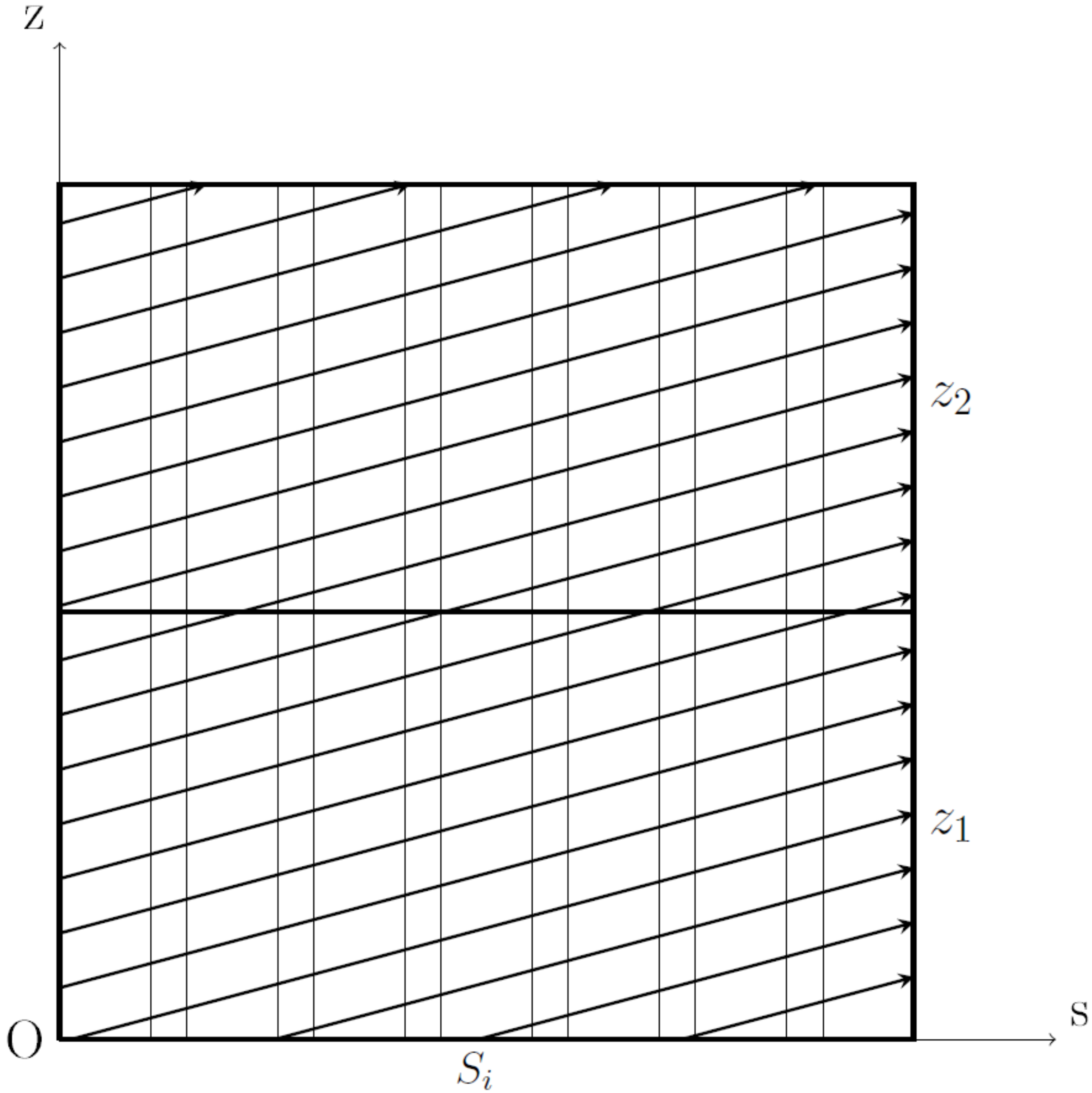
FINAL CONSIDERATIONS

- Higher numbers of floating point operations per chord
- Some of them can be vectorized



FINAL CONSIDERATIONS

- Higher numbers of floating point operations per chord
- Some of them can be vectorized
- Less axial planes also means less chords (~ -15%)
- Less memory needed



POLYNOMIAL VS STEP: RESULTS 2

Method	Step					Polynomial (N_p=2)		
Δr (cm)	0.05					0.05		
Δs (cm)	1.0					1.0		0.5
Axial meshes	**57**	**110**	**180**	**257**	**600**	**5**	**6**	**12**
# chords	29.57 M	31.88 M	34.93 M	38.29 M	53.24 M	27.31 M	27.35 M	55.24 M
# classes	17.07 M	21.12 M	21.80 M	20.33 M	18.16 M	1.90 M	2.28 M	4.57 M
Classification	83.82 %	74.67 %	66.68 %	59.56 %	52.39 %	98.4 %	98.09 %	96.21 %
Self-shielding	**Tone:**							
k_{eff}	1.163761	1.165075	1.165412	1.165672	1.165744	1.165584	1.165805	1.165801
ρ **err/T4 (PCM)**	**-96.45**	**+0.44**	**+25.43**	**+44.66**	**+49.72**	**+37.76**	**+54.00**	**+53.92**
Time	12 210 s	22 719 s	37 785 s	55 472 s	127 809 s	15 384 s	16 326 s	34 775 s
Self-shielding	**Sub-Groups:**							
k_{eff}	1.164114	1.16543	1.165767	1.166027	1.166094	1.165927	1.166147	1.166154
ρ **err/T4 (PCM)**	**-70.54**	**+26.38**	**+ 51.22**	**+70.28**	**+75.42**	**+ 63.01**	**+ 79.27**	**+79.72**
Time	12 575 s	24 031 s	38 454 s	56 500 s	124 470 s	16 174 s	17 316 s	50 620 s

- An impressive gain in computational meshes is obtained

SELF-SHIELDING/ACCELERATION IMPACT

Self-shielding effect		
	k_{eff}	δk_{eff}
NO self-shielding	1.093190	-2932 PCM
Sub-Groups method	1.127149	+84 PCM
Tone method	1.126910	+62 PCM

Table 5: Self-shielding effect for the full-column case in nominal conditions. δk_{eff} refers to the relative error (in PCM) with respect to the reference Tripoli4 calculation. Nominal conditions.

Acceleration effectiveness			
Acc./Free	DP_1 polynomial order		
	0	1	2
Time	0.16	–	0.06
Outers	0.66	–	0.11
Inners	0.11	–	0.02
Memory	1.96	–	10.06

Table 6: Ratios of times, number of iterations and memory footprint between accelerated calculations and free iterations for varying order of the spatial polynomial order of the DP_1 operator. The case considered is the full-column assembly in nominal conditions.

- A factor 20 of computational time reduction can be obtained but there is a memory price to pay. (Work on it is under way!)
- NOTE: All micro/macro-scopic reaction rate errors are below 1%

CONCLUSIONS

- Polynomial MOC is on the way
- Classifications of chords is of fundamental importance
- Dpn acceleration works but it is memory expensive
- Many ways are possible for memory reduction
- How about XS?

REFERENCES

1. Sanchez, R. (2012), `Prospects in deterministic three-dimensional whole-core transport calculations' Nuclear Engineering and Technology 44(5), 113-150.

2. W. Boyd, A. Siegel, S. He, B. Forget and K. Smith: "Parallel performance results for the OpenMOC method of characteristic code on multi-core platforms", http://dx.doi.org/:10.1177/1094342016630388 International Journal of High Performance Computing Applications, February 15, 2016.

3. D.Sciannandrone, , S. Santandrea, R.Sanchez: "Optimized tracking strategies for step MOC calculations in extruded 3D axial geometries", Ann. Nucl. Energy Vol.87 49-60 (2016) http://dx.doi.org/10.1016/j.anucene.2015.05.014.

4. Santandrea S., Sciannandrone D., Sanchez R., Mao L. and Graziano L.: " A neutron transport characteristics method for 3D axially extruded geometries coupled with a fine group self-shielded environment", published in NSE 2017

5. Santandrea S., Graziano L & Sciannandrone D.: "Accelerated Polynomial axial expansions for full 3D neutron transport MOC in the APOLLO3R code system as applied to the ASTRID fast breeder reactor » published ANE 2018

MOC 1 : BASIC ITERATIVE SCHEME

- *scattering term expansion* $$q(\mathbf{r},\Omega) \sim \sum_i q_i(\Omega)\theta_i(\mathbf{r})$$

$$q_i(\Omega) = \underbrace{\sum_{k=0}^{K} \Sigma_{sk,i} \sum_{l=-k}^{k} \phi^l_{k,i} A^l_k(\Omega)}_{scattering} \quad + \quad \underbrace{S_i(\Omega)}_{external\ source}$$

- *cell averaged angular flux moments*

$$\phi^l_{k,i} = \frac{1}{4\pi}\int_{(4\pi)} d\Omega A^l_k(\Omega)\psi_i(\Omega) \sim \sum_n w_n A^l_k(\Omega_n)\psi_i(\Omega_n)$$

- *positive method*
- *no fix-up is necessary*
- *arbitrary anisotropy order*

DP_N : SYNTHETIC ACCELERATION

Synthetic acceleration

- *perform a free iteration*

$$\begin{pmatrix} \Phi^{(n)} \\ J_+^{(n)} \end{pmatrix} \rightarrow \begin{pmatrix} \Phi_{free}^{(n+1)} \\ J_{+,free}^{(n+1)} \end{pmatrix}$$

- *solve synthetic acceleration for*

$$D\begin{pmatrix} \delta\Phi^{(n)} \\ \delta J_+^{(n)} \end{pmatrix} = H\begin{pmatrix} \Phi_{free}^{(n+1)} - \Phi_{free}^{(n)} \\ J_{+,free}^{(n+1)} - J_{+,free}^{(n)} \end{pmatrix}$$

- *correct free iteration values*

$$\Phi_{acce}^{(n+1)} = \Phi_{free}^{(n+1)} + \delta\Phi$$

$$J_{+,acce}^{(n+1)} = J_{+,free}^{(n+1)} + \delta J_+$$

- *this approach can be extended from inhomogeneous to eigenvalue problems*

DP_N 2

- *transmission and escape first flight probabilities (symmetry and conservation preserved by numerical scheme) for SC method:*

$$T_{\alpha\beta}^{\rho\upsilon} = T_{\beta\alpha}^{\rho\upsilon} = \int_{\alpha} dS \int_{\beta\to\alpha} d\Omega A^{\rho}\left(\mathbf{\Omega}\right) A^{\upsilon}\left(\mathbf{\Omega}\right) |\mathbf{n}\cdot\mathbf{\Omega}|\, e^{-\Sigma_i R(\mathbf{r},\mathbf{\Omega})} \qquad (symmetry)$$

$$E_{\alpha}^{\rho\upsilon} = \frac{1}{\Sigma_i V_i}\left(A_{\alpha}^{\rho\upsilon} - \sum_{\alpha\in\partial i} T_{\alpha\beta}^{\rho\upsilon}\right) \qquad (conservation)$$

- *similar formulas can be written for the Linear Surface method.*

- *numerical evaluation (coherence with transport)*

$$T_{\alpha\beta}^{\rho\upsilon} \sim \sum_{\mathbf{\Omega}} \mathrm{w}_{\mathbf{\Omega}} \mathrm{A}^{\rho}\left(\mathbf{\Omega}\right) \mathrm{A}^{\upsilon}\left(\mathbf{\Omega}\right) \sum_{(t,\mathbf{\Omega})\in\beta\to\alpha} \mathrm{w}_{\perp}(t,\mathbf{\Omega}) e^{-\Sigma_i R(t,\mathbf{\Omega})}$$

DP_N 3

- *after elimination of cell fluxes, the DP_N acceleration equations are solved iteratively for the currents*

$$\vec{J}_{+,\alpha} = \sum_{\beta \in \partial i} \hat{T}\,\vec{J}_{-,\beta} + \vec{J}_S$$

$$\hat{T}^{\rho\upsilon}_{\alpha\beta} = T^{\rho\upsilon}_{\alpha\beta} + E^{\rho 0}_{\alpha} \frac{\Sigma_i V_i \Sigma_{si}}{\Sigma_{ai} + E^{00}\Sigma_{si}} E^{0\upsilon}_{\beta} S^{\upsilon} \qquad \textit{generalized transmission related to multicollisional processes}$$

- *solution with a Krilov iterator (BCGS or GMRES):* $M\,\vec{\psi} = \vec{S}$

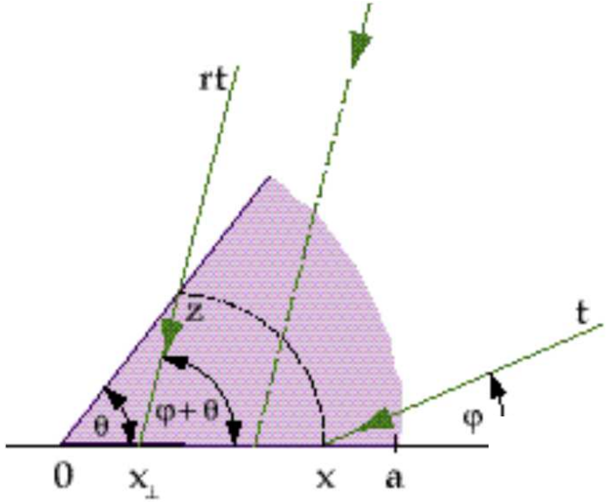
$$iterator: \quad M = 1 - \hat{T}$$

with an adapted ILU0 and domain decomposition method.

MOC 3 : BOUNDARY CONDITIONS

Adapted tracking is done to exactly take into account symmetries and boundary conditions

p / 2 rotation



Cyclic tracking for

infinite periodic systems