A hypothesis or theory is clear, decisive, and positive, but it is believed by no one but the person who created it. Experimental findings, on the other hand, are messy, inexact things, which are believed by everyone except the person who did that work.

Harlow Shapley
Through Rugged Ways to the Stars

UC HIPACC Summer School 22Jul2011

Reaction Networks

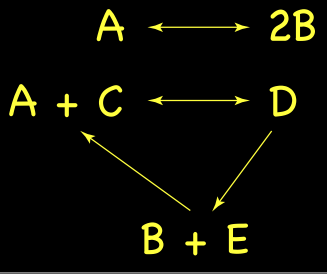
Frank Timmes
School of Earth and Space Exploration
Advanced Computing Center
Arizona State University

cococubed.asu.edu ftimmes@asu.edu

Reaction networks are an key tool for modeling nucleosynthesis processes and their associated energy generation.

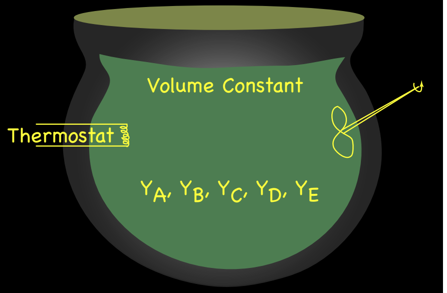
In essence, nuclear reaction networks in astrophysics are a system of nonlinear, stiff, generally sparse, ordinary differential equations (ODEs).

Let's walk through an example of a reaction network and indicate informally how it induces a system of ODEs.



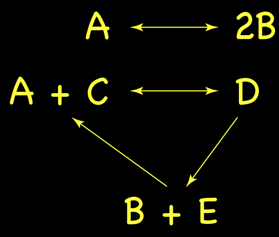
Suppose we throw the various species in a pot that is constantly stirred so its contents remain spatially homogeneous for all time.

We'll also assume that the contents are kept at constant temperature and volume (constant density). For historical reasons this is referred to as hydrostatic burning in astrophysics.



Denote the instantaneous values of the molar abundances by Y_A, Y_B, Y_C, Y_D, and Y_E. We want to write down five ODEs that describe the evolution of the five mole fractions.

Consider first the instantaneous rate of change of Y_A.



Every time A→2B we lose one unit of A and this reaction occurs with an instantaneous, non-negative, real valued rate of $K_{A\to 2B}$.

Similarly the reaction A + C→D loses a unit of species A, while 2B→A, B+E→A+C, and D→A+C each produce a unit of species A. So we write

$$\dot{Y}_A = -K_{A\to 2B} + K_{2B\to A} - K_{A+C\to D} + K_{D\to A+C} + K_{B+E\to A+C}$$

Continuing in this way, we form a system of ODEs that govern our reactor:

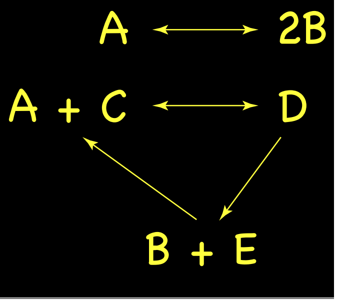
$$\dot{Y}_A = -K_{A\to 2B} + K_{2B\to A} - K_{A+C\to D} + K_{D\to A+C} + K_{B+E\to A+C}$$

$$\dot{Y}_B = 2K_{A\to 2B} - 2K_{2B\to A} + K_{D\to B+E} - K_{B+E\to A+C}$$

$$\dot{Y}_C = -K_{A+C\to D} + K_{D\to A+C} + K_{B+E\to A+C}$$

$$\dot{Y}_D = K_{A+C\to D} - K_{D\to A+C} - K_{D\to B+E}$$

$$\dot{Y}_E = K_{D\to B+E} - K_{B+E\to A+C}$$



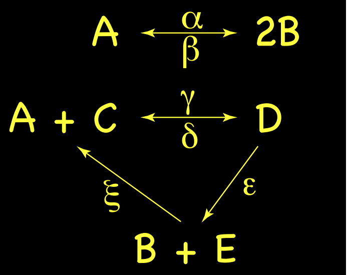
For A→2B, the more A there is, the more reaction there will be. We take the rate of A→2B to be proportional to Y_A : $K_{A\to 2B} = \alpha Y_A$.

For A+C→D, a unit of species A must meet a unit of species C. We take the probability of such an encounter to be proportional to the product $Y_A Y_C$: $K_{A+C\to D} = \gamma Y_A Y_C$.

$\mathcal{K}_{\mathrm{A\to 2B}} = \alpha Y_A$ $\quad$ $\mathcal{K}_{\mathrm{2B\to A}} = \beta Y_B^2$

$\mathcal{K}_{\mathrm{A+C\to D}} = \gamma Y_A Y_C$ $\quad$ $\mathcal{K}_{\mathrm{D\to B+E}} = \varepsilon Y_D$

$\mathcal{K}_{\mathrm{D\to A+C}} = \delta Y_D$ $\quad$ $\mathcal{K}_{\mathrm{B+E\to A+C}} = \xi Y_B Y_E$



The functions α, β, γ, δ, ε, and ξ may depend on temperature and density.

And our reaction network takes the final form

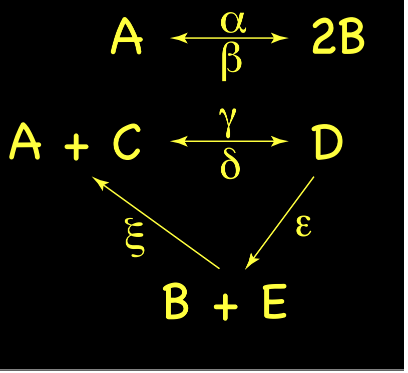
$$\dot{Y}_A = -\alpha Y_A + \beta Y_B^2 - \gamma Y_A Y_C + \delta Y_D + \xi Y_B Y_E$$

$$\dot{Y}_B = 2\alpha Y_A - 2\beta Y_B^2 + \epsilon Y_D - \xi Y_B Y_E$$

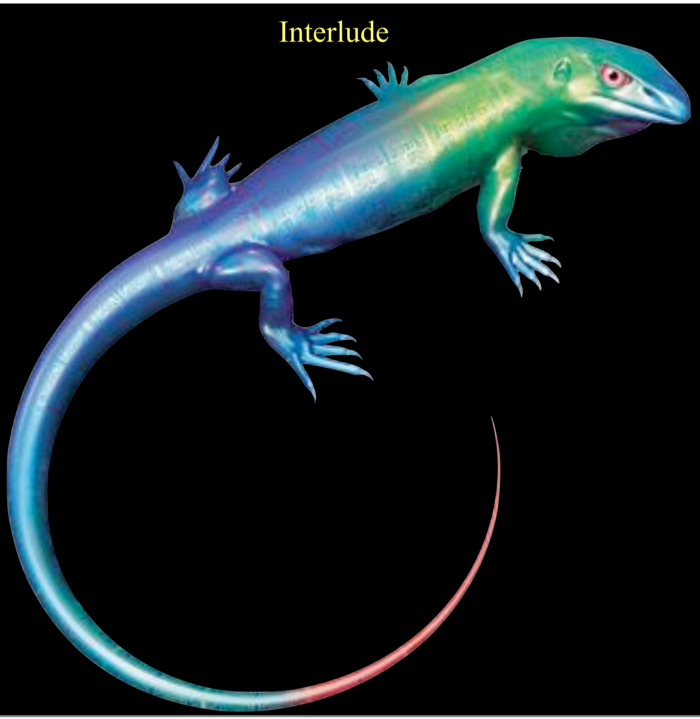
$$\dot{Y}_C = -\gamma Y_A Y_C + \delta Y_D + \xi Y_B Y_E$$

$$\dot{Y}_D = \gamma Y_A Y_C - \delta Y_D - \epsilon Y_D$$

$$\dot{Y}_E = \epsilon Y_D - \xi Y_B Y_E$$



Interlude



Nuclear reactions involve three of the four fundamental forces, and involve the emission or absorption of nuclei and nucleons, photons (γ-rays) and leptons (e-,ν, and their anti-particles).

Weak interactions (those involving leptons) generally proceed more slowly than those involving nucleons and photons, but these are the only reactions that can change the global proton to neutron ratio.

A key quantity is the cross section σ for a nuclear reaction.σ_{ij} for the reaction *i(j,k)l* is the number of reactions per second on target nucleus *i* divided by the flux of nuclei of type *j* (number/cm^2/s).

$$\sigma(v) = \frac{\text{number of reactions per sec}}{\text{flux of incoming projectiles}} = \frac{r_{ij}/n_i}{n_j v_{ij}}$$

Nuclear cross sections are usually reported in “barns”, 10^{-24} cm^2.

The reaction rate per unit volume r_{ij}, in the simplest case, is then

$$r_{ij} = [\text{flux of } j] n_i \sigma_{ij}(v) = v_{ij} n_j n_i \sigma_{ij}(v) \quad \text{cm}^{-3}\text{s}^{-1}$$

More generally, the targets and projectiles have distributions of velocities, in which case r_{ij} is given by

$$r_{i,j} = \int \sigma(|\vec{v}_i - \vec{v}_j|)|\vec{v}_i - \vec{v}_j| d^3n_i d^3n_j \quad \text{cm}^{-3}\text{s}^{-1}$$

Evaluation of the integrals depends on the particle statistics. For nuclei i and j that obey Maxwell–Boltzmann statistics

$$d^3n = n \left(\frac{m}{2\pi k_B T}\right)^{3/2} \exp\left(-\frac{mv^2}{2k_B T}\right) d^3v$$

n$_i$ and *n*$_j$ can be moved outside of the integral.

$$r_{ij} < \sigma v >_{ij} n_i n_j = (N_A \rho)^2 < \sigma v >_{ij} Y_i Y_j \quad \text{cm}^{-3}\text{s}^{-1}$$

where <σv>$_{ij}$ is the velocity integrated cross section. The rate of change in the number density of species i with time is

$$\dot{n}_i = \sum_{j,k} r_{jk} \quad \text{cm}^{-3}\text{s}^{-1}$$

or

$$\dot{Y}_i = \sum_{j,k} N_A \rho < \sigma v >_{ij} Y_j Y_k = \sum_{j,k} \lambda_{ij} \rho Y_j Y_k = \sum_{j,k} R_{ij} Y_j Y_k \quad \text{s}^{-1}$$

where λ_{ij} is what common reaction rate compilations list, and R_{ij} is “the reaction rate” used in my codes.

Consider a unidirectional binary reaction with unity coefficients.

$$i + j \rightarrow k + l$$

$$\dot{Y}_i = -Y_i Y_j R_{ij}$$

$$\dot{Y}_j = -Y_i Y_j R_{ij}$$

$$\dot{Y}_k = Y_i Y_j R_{ij}$$

$$\dot{Y}_l = Y_i Y_j R_{ij}$$

Where the reaction rate R_{ij} absorbs the density, Avogado number, and $<\sigma v>_{ij}$ terms.

Consider the unidirectional case when the coefficients are not unity.

$$c_i i + c_j j \rightarrow c_k k + c_l l$$

$$\dot{Y}_i = -\frac{c_i}{c_i! c_j!} Y_i^{c_i} Y_j^{c_j} R_{ij}$$

$$\dot{Y}_j = -\frac{c_j}{c_i! c_j!} Y_i^{c_i} Y_j^{c_j} R_{ij}$$

$$\dot{Y}_k = \frac{c_k}{c_i! c_j!} Y_i^{c_i} Y_j^{c_j} R_{ij}$$

$$\dot{Y}_l = \frac{c_l}{c_i! c_j!} Y_i^{c_i} Y_j^{c_j} R_{ij}$$

For identical reactants in the entrance channel, i=j, set $c_i = 2c_i$ and $c_j = 0$.

For a general bidirectional binary reaction

$$c_i i + c_j j \leftrightarrow c_k k + c_l l$$

$$\dot{Y}_p = \sum_{r,s} \frac{c_p}{c_r! c_s!} Y_r^{c_r} Y_s^{c_s} R_{rs} - \sum_q \frac{c_p}{c_p! c_q!} Y_p^{c_p} Y_q^{c_q} R_{pq}$$

If there are identical reactants, i=j, set $c_i = 2c_i$ and $c_q = c_s = 0$.

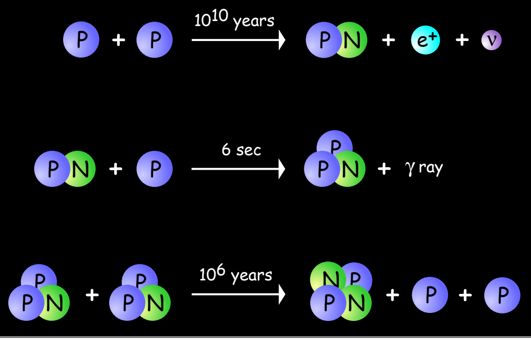
Interlude

The Medical Alchemist
Franz Christoph Janneck,
(1703 - 1761)
Oil on copper - 13" x 9"

Time Integration

The ODEs from nuclear reaction networks are nonlinear and stiff.

Physically a stiff system of ODEs means some isotopes are changing on much faster timescales than other isotopes. The hydrogen burning PPI chain is an excellent example.



Mathematically, stiffness means the ratio of the maximum to the minimum eigenvalues λj of the Jacobian matrix are large.

$$\mathcal{S} = \frac{\max|\mathrm{Re}(\lambda_j)|}{\min|\mathrm{Re}(\lambda_j)|} \gg 1$$

S > 10^{15} is not uncommon in nuclear astrophysics.

Pragmatically, stiffness means that an implicit time integration is typically necessary to solve the initial value problem.

This, in turn, means we'll be solving systems of linear equations. As the linear algebra will generally dominate the time to solution, we'll want to use efficient solvers.

Our system of ODEs

$$\dot{Y}_i = \sum_j C_i R_j Y_j + \sum_{jk} \frac{C_i}{C_j! C_k!} R_{jk} Y_j Y_k + \sum_{jkl} \frac{C_i}{C_j! C_k! C_l!} R_{jk} Y_j Y_k Y_l$$

may be written in vector form as

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y})$$

The Jacobian matrix is the derivative of the ODEs with respect to their dependent variables

$$\tilde{J} = \frac{\partial \mathbf{f}}{\partial \mathbf{y}}$$

Let's do an example of forming the Jacobian matrix ...

Consider the ^{12}C(α,γ)^{16}O reaction proceeding at a rate R.

$$\dot{Y}(^4\mathrm{He}) = -Y(^4\mathrm{He})\ Y(^{12}\mathrm{C})\ R + \ldots$$
$$\dot{Y}(^{12}\mathrm{C}) = -Y(^4\mathrm{He})\ Y(^{12}\mathrm{C})\ R\ + \ldots$$
$$\dot{Y}(^{16}\mathrm{O}) = +Y(^4\mathrm{He})\ Y(^{12}\mathrm{C})\ R\ + \ldots$$

Each right hand side contributes two Jacobian matrix elements:

$$J(^4\mathrm{He}, ^4\mathrm{He}) = \partial \dot{Y}(^4\mathrm{He}) / \partial Y(^4\mathrm{He}) = -Y(^{12}\mathrm{C})\ R\ + \ldots$$

$$J(^4\mathrm{He}, ^{12}\mathrm{C}) = \partial \dot{Y}(^4\mathrm{He}) / \partial Y(^{12}\mathrm{C}) = -Y(^4\mathrm{He})\ R\ + \ldots$$

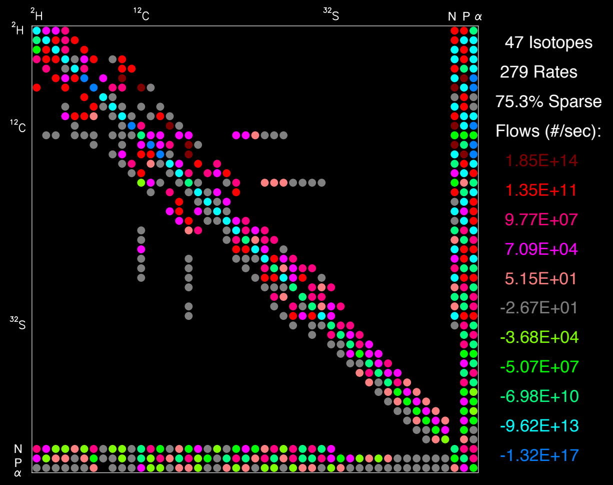
$$J(^{12}\mathrm{C}, ^4\mathrm{He}) = \partial \dot{Y}(^{12}\mathrm{C}) / \partial Y(^4\mathrm{He}) = -Y(^{12}\mathrm{C})\ R\ + \ldots$$

$$J(^{12}\mathrm{C}, ^{12}\mathrm{C}) = \partial \dot{Y}(^{12}\mathrm{C}) / \partial Y(^{12}\mathrm{C}) = -Y(^4\mathrm{He})\ R\ + \ldots$$

$$J(^{16}\mathrm{O}, ^4\mathrm{He}) = \partial \dot{Y}(^{16}\mathrm{O}) / \partial Y(^4\mathrm{He}) = +Y(^{12}\mathrm{C})\ R\ + \ldots$$

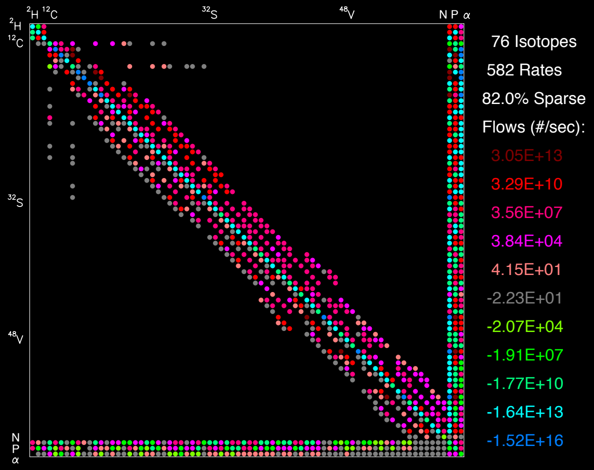
$$J(^{16}\mathrm{O}, ^{12}\mathrm{C}) = \partial \dot{Y}(^{16}\mathrm{O}) / \partial Y(^{12}\mathrm{C}) = +Y(^4\mathrm{He})\ R\ + \ldots$$

The Jabobian matrix elements represent flows into (positive) or out of (negative) an isotope.



The matrix is not positive-definite or symmetric as reactions rates are not usually equal, but they are typically diagonally dominant.

For a constant number of isotopes, the pattern of nonzeros doesn't change with time, but each matrix element may change in magnitude or sign as the temperature, density, or abundances change with time.



The matrices get sparser as the number of isotopes increase.

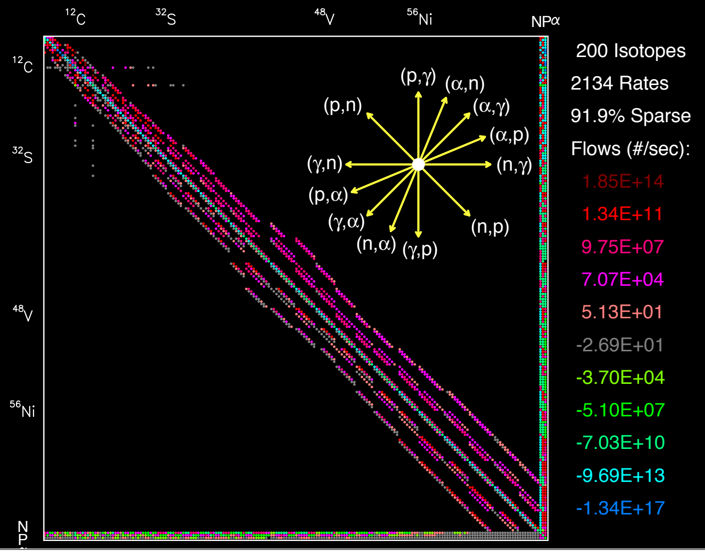
In principal every species reacts with every other species, resulting in a full, dense Jacobian matrix. In practice it is possible to neglect most of these reactions.

Captures of n, p, d, t, ^{3}He and α on heavy nuclei are easier than fusions of heavier nuclei because of the Z_i Z_j dependence of the repulsive Coulomb term in the nuclear potential.

With the exception of the PP-chains and Big Bang nucleosynthesis, reactions involving d, t, and ^{3}He are negligible because their abundances are effectively zero.

Photodisintegration reactions tend to eject free nucleons or α-particles instead of splitting the nucleus into bigger pieces.

Thus, with a few important exceptions, we only need to consider twelve reactions linking a nucleus to its neighbors by the capture of an n, p, α or γ and release a different one of these four.



Consider the ODE

$$\dot{y} = -y \longrightarrow y(t) = y(0)e^{-t}$$

In explicit methods, the state at a future time is computed as a function of the state at the current time:

$$\frac{\Delta y}{\Delta t} = \frac{y(t+\Delta t) - y(t)}{\Delta t} = -y(t)$$

$$y(t+\Delta t) = y(t)(1-\Delta t) \longrightarrow -\infty \text{ as } \Delta t \to \infty \qquad |1-\Delta t| < 1$$

In implicit methods, the state at the next time instant is computed as a function of the state at the next time instant:

$$\frac{\Delta y}{\Delta t} = \frac{y(t+\Delta t) - y(t)}{\Delta t} = -y(t+\Delta t)$$

$$y(t+\Delta t) = \frac{y(t)}{1+\Delta t} \longrightarrow 0 \text{ as } \Delta t \to \infty \qquad |1+\Delta t| > 1$$

Given the initial conditions (temperature, density, composition), we wish to evolve the stiff ODEs that represent our reaction network.

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y})$$

We'll take a look at several (but not all) implicit methods Nearly all of these methods are in active use among various researchers.

A good ODE integrator should exert some adaptive control over its own progress, making frequent changes in its stepsize.

The purpose adaptive stepsize control is to achieve a predetermined accuracy in the solution with minimum computational effort.

Many small steps should tiptoe through treacherous terrain,
while a few great strides should speed through smooth countryside.

The resulting gains in efficiency are not mere 10%s or factors of 2; t hey can be factors of 10, a 100, or more.

Sometimes accuracy may be demanded not directly in the solution itself, but in some conserved quantity that can be monitored.

Implementation of adaptive stepsize control requires that the stepping algorithm return information about its performance, and most importantly, an estimate of its truncation error.

The simplest 1rst order Euler method advances over a time step h by

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta$$

where the change Δ is found by expanding $\mathbf{f}(\mathbf{y}_{n+1})$ about $\mathbf{f}(\mathbf{y}_n)$

$$(\tilde{\mathbf{1}}/h - \tilde{\mathbf{J}}) \cdot \Delta = \mathbf{f}(\mathbf{y}_n)$$

which is simply

$$\tilde{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b}$$

This method costs 1 Jacobian and 1 right-hand side evaluation,
1 matrix reduction, and 1 backsubstitution.

This is the smallest possible cost per time step and is one of the most common methods for evolving nuclear reaction networks.

This method is 1rst order accurate, which as given provides no rigorous estimate of the truncation error over a given time step.

Heuristics, usually limiting the change in any abundance to be less than some small percentage, are often invoked to gain some sense of accuracy and to form the next time step.

One could implement “step doubling” to gain a formal accuracy estimate; take two half step and one full step. If the two solutions agree within some specified accuracy tolerance, accept the time step.

Step doubling is relatively expensive. Higher order methods obtain accuracy estimates by comparing solutions at different orders. We turn to two of these schemes next.

The 4th order Kaps-Rentrop method advances a time step h by

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \sum_{i=1}^{4} b_i \Delta_i$$

where the Δ_i are found from solving the staged equations

$$\tilde{\mathbf{A}} = (\tilde{\mathbf{1}}/\gamma h - \tilde{\mathbf{J}})$$

$$\tilde{\mathbf{A}} \cdot \Delta_1 = \mathbf{f}(\mathbf{y}_n)$$

$$\tilde{\mathbf{A}} \cdot \Delta_2 = \mathbf{f}(\mathbf{y}_n + a_{21}\Delta_1) + c_{21}\Delta_1/h$$

$$\tilde{\mathbf{A}} \cdot \Delta_3 = \mathbf{f}(\mathbf{y}_n + a_{31}\Delta_1 + a_{32}\Delta_2) + (c_{31}\Delta_1 + c_{32}\Delta_2)/h$$

$$\tilde{\mathbf{A}} \cdot \Delta_4 = \mathbf{f}(\mathbf{y}_n + a_{41}\Delta_1 + a_{42}\Delta_2 + a_{43}\Delta_3) + (c_{41}\Delta_1 + c_{42}\Delta_2 + c_{43}\Delta_3)/h$$

The b_i, γ, a_{ij}, and c_{ij} are fixed constants of the method.

The truncation error is estimated by comparing an embedded 3rd-order solution with the 4th-order solution.
This in turn, permits adaptive stepsize control.

This method costs 1 Jacobian, 3 right-hand side evaluations, 1 matrix reduction, and 4 backsubstitutions for a time step that meets the specified integration accuracy.

In this method not all of the right-hand sides are known in advance, Δ_4 depends on Δ_3 ... depends on Δ_1.

This general feature of higher-order integration methods impacts the optimal choice of a linear algebra package.

The variable order Bader-Deuflhard method advances a large time step H from y_n to y_{n+1} by forming

$$h = H/m \qquad \tilde{\mathbf{A}} = (\tilde{\mathbf{1}} - \tilde{\mathbf{J}})$$

$$\tilde{\mathbf{A}} \cdot \Delta_0 = h\mathbf{f}(\mathbf{y}_n) \qquad \mathbf{y}_1 = \mathbf{y}_n + \Delta_0$$

then for k=1,2 ... m-1 solving the staged equations

$$\tilde{\mathbf{A}} \cdot \mathbf{x} = h\mathbf{f}(\mathbf{y}_k) - \Delta_{k-1}$$

$$\Delta_k = \Delta_{k-1} + 2\mathbf{x}$$

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \Delta_k$$

closure is obtained by the last stage

$$\tilde{\mathbf{A}} \cdot \Delta_m = h[\mathbf{f}(\mathbf{y}_m) - \Delta_{m-1}]$$

$$\mathbf{y}_{n+1} = \mathbf{y}_m + \Delta_m$$

The staged sequence is executed at least twice, yielding a 5th order method, or a maximum 7 times, yielding a 15th order method.

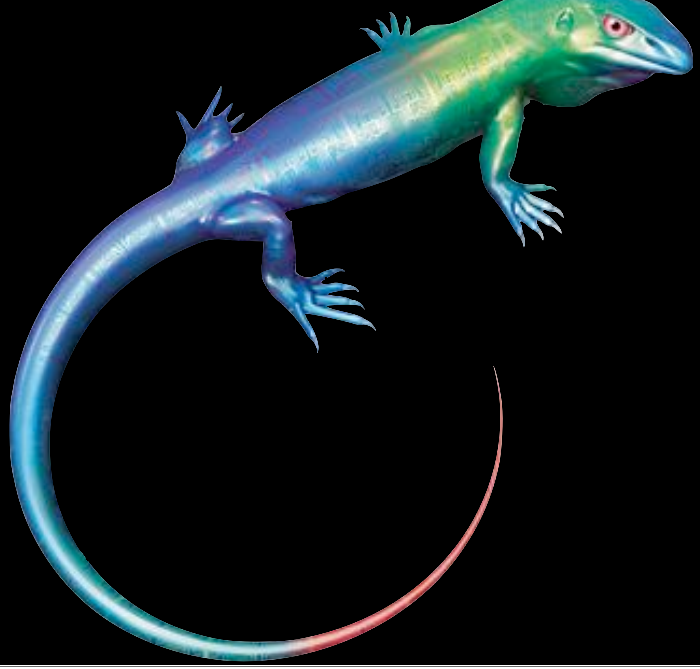
The exact number of times the staged sequence is executed depends on the accuracy requirements and the smoothness of the solution.

Accuracy estimates of a time step is made by comparing solutions from different orders. This in turn permits adaptive stepsize control.

Minimum cost is 1 Jacobian, 8 right-hand side evaluations, 2 matrix reductions, and 10 backsubstitutions for a time step that meets the specified integration accuracy.

Cost per step is at least twice as large as the Euler or Kaps-Rentrop methods, but it may be more efficient if time steps at least twice as large can be taken.

Interlude



Ã·x = b

Matrix Ã is reduced to upper triangular form in tandem with a right-hand side b by Gaussian elimination, and backsubstitution on the upper triangular matrix yields the solution to Ã·x = b.

$$\begin{bmatrix} 2 & -1 & 1 \\ -2 & 2 & -3 \\ 2 & -4 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ -7 \\ 3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1/2 & 1/2 \\ -2 & 2 & -3 \\ 2 & -4 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3/2 \\ -7 \\ 3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1/2 & 1/2 \\ 0 & 1 & -2 \\ 0 & -3 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3/2 \\ -4 \\ 0 \end{bmatrix}$$

This is the method you probably first learned.

$$\begin{bmatrix} 1 & -1/2 & 1/2 \\ 0 & 1 & -2 \\ 0 & 0 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3/2 \\ -4 \\ -12 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1/2 & 1/2 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3/2 \\ -4 \\ 3 \end{bmatrix}$$

If the arithmetic is exact, then the answer computed in this manner will be exact, if no zeros appear on the diagonal.

$$x_3 = 3 \quad x_2 = 2 \quad x_1 = 1$$

But computer arithmetic is not exact, so there will always be some truncation and rounding error in the answer.

If a “small” number appears on the diagonal, then its use as the pivot may lead to computing differences between big numbers and little numbers with a subsequent loss of precision.

A way around this problem is to ensure small pivots are not used by swapping rows (partial pivoting) or rows and columns (full pivoting), so as to use a particularly desirable pivot element.

What is a desirable pivot? It is not completely known theoretically. It is known, both theoretically and in practice, that simply picking the largest available element as the pivot is a very good choice.

LEQS is a dense matrix, Gaussian elimination routine.

The maximum element in each row serves as the pivot element, but no row or column interchanges are performed, so LEQS may be unstable on matrices that are not diagonally dominant.

A tiny amount of effort is expended to minimize calculations with matrix elements that are zero.

All Gaussian elimination routines have the disadvantage that for a sequence of right-hand sides, the entire matrix must be decomposed for each right-hand side.

The origin of LEQS is somewhat obscure, but circa 1962. It may be the most common solver used for evolving nuclear reaction networks.

Ford-Seattle
1962

A frequently used form of Gauss Elimination is LU decomposition.

The basic idea is to find two matrices L and U such that $LU = \tilde{A}$, where L is a lower triangular matrix (zero above the diagonal) and U is an upper triangular matrix (zero below the diagonal).

$$\tilde{A}\cdot\mathbf{x} = \left(\tilde{L}\cdot\tilde{U}\right)\cdot\mathbf{x} = \tilde{L}\cdot\left(\tilde{U}\cdot\mathbf{x}\right) = \mathbf{b}$$

Once we have computed L and U we the solve L·y=b then U·x=y, a process that takes $O(n^2)$ operations.

While the factorization stage still requires $O(n^3)$ operations, it need be done only once. We can solve with as many right-hand sides as we care to, one at a time. This is an advantage.

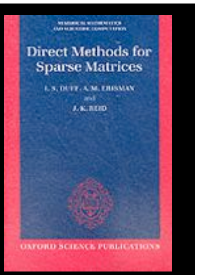
A linear system is called sparse if only a relatively small number of its matrix elements a_{ij} are nonzero.

It is wasteful to use general methods on such problems, because most of storage the $O(n^3)$ operations involve zero operands.

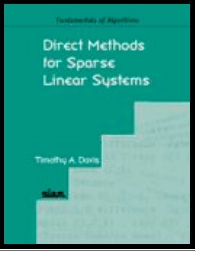
Direct methods for sparse matrices are not that different from dense LU decomposition methods; they are just cleverly applied with due attention to the bookkeeping of zero elements.

The basic approach that all solvers use are
1) Symbolic decomposition
2) Numerical decomposition
3) Backsubstitution
4) Iterative polishing

MA28 is the Coke classic of sparse matrix solvers.
hsl.rl.ac.uk/archive/hslarchive.html


Direct Methods for Sparse Matrices
I. S. DUFF · A. M. ERISMAN
and
J. K. REID
OXFORD SCIENCE PUBLICATIONS

UMFPACK is a modern, direct sparse matrix solver.
www.cise.ufl.edu/research/sparse/umfpack


Fundamentals of Algorithms
Direct Methods for Sparse Linear Systems
Timothy A. Davis
siam

Iterative, or “matrix free” methods for sparse systems only reference the matrix Ã through multiplication of a vector.

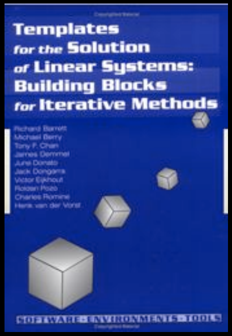
Iterative methods can be slow to converge and the number of iterations to reach a given level of accuracy is not known a priori.

A popular method, generalized minimum residuals, seeks a minimization of the function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{r}\cdot\mathbf{r} = \frac{1}{2}\left|\tilde{\mathbf{A}}\cdot\mathbf{x}-\mathbf{b}\right|^2 \qquad \nabla f(\mathbf{x}) = \tilde{\mathbf{A}}^T\cdot\left(\tilde{\mathbf{A}}\cdot\mathbf{x}-\mathbf{b}\right)$$

One way to generate a good “guess” is to solve some portion of Ã, call it matrix Z, that is easy to solve. Z is called the preconditioner.

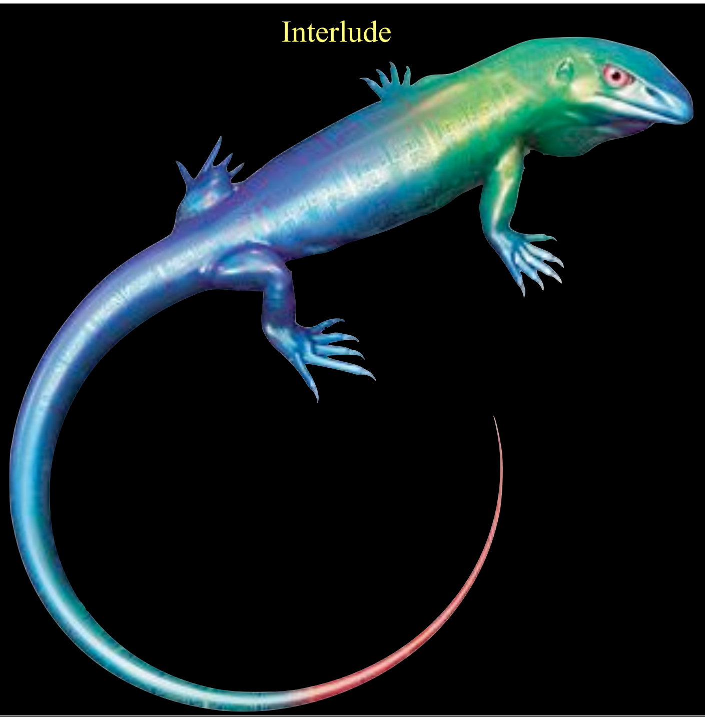
BiCG is described by Barret et al in “Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods”.
netlib2.cs.utk.edu/linalg/html_templates/Templates.html



SPARSKIT Version 2.

SPARSKIT is a modern, iterative sparse matrix solver.
www-users.cs.umn.edu/~saad/software/SPARSKIT/sparskit.html

Interlude



Energy Generation

An important consequence of changing the composition is the release (or absorption) of energy. The energy generation rate is

$$\dot{\epsilon}_{\rm nuc} = -\sum_i N_A M_i c^2 \dot{Y}_i - \dot{\epsilon}_\nu \quad ({\rm erg\ g^{-1}\ s^{-1}})$$

where $M_i c^2$ is the rest mass energy of species i. Using

$$M_i = A_i m_\mu + M_{{\rm ex},i}$$

$$M_{ex,i} = Z_i M_{ex,p} + N_i M_{ex,n} - B_i/c^2$$

the energy generation rate is sometimes written as

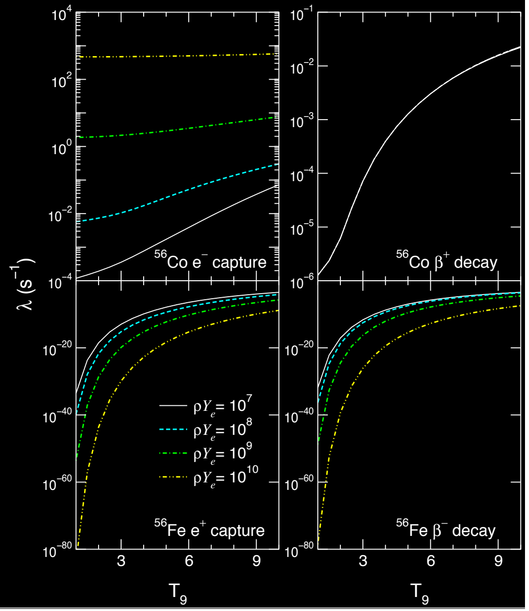
$$\dot{\epsilon}_{\rm nuc} = \sum_i N_A E_{{\rm bind},i} \dot{Y}_i - \sum_i N_A (Z_i M_{{\rm ex},p} + N_i M_{{\rm ex},n}) \dot{Y}_i - \dot{\epsilon}_\nu$$

The energy lost to the freely streaming neutrinos has two parts: weak reactions and neutrino thermal processes.

For weak reactions, average energy losses are calculated for each nucleus by considering the excited state distribution, Gamow-Teller distribution, etc.

The results are tabulated; see for example Langanke & Martinez-Pinedo (2000).

$$\dot{\epsilon}_\nu = \sum_i \langle E_\nu \rangle \dot{Y}_{i,weak}$$



Thermal neutrino processes include

pair neutrino: $e^+ + e^- \rightarrow \nu + \bar{\nu}$

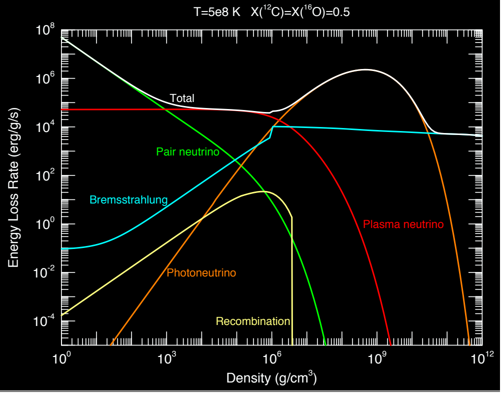
photoneutrino: $e^+ + \gamma \rightarrow e^- + \nu + \bar{\nu}$

plasma neutrino: $\gamma_{plasmon} \rightarrow \nu + \bar{\nu}$

bremsstrahlung: $e^- + N(Z,A) \rightarrow e^- + N(Z,A) + \nu + \bar{\nu}$

recombination: $e^-_{continuum} \rightarrow e^-_{bound} + \nu + \bar{\nu}$

Results are typically expressed in tables or fitting formulas, for example, Itoh et al. 1996.



Alpha-chain networks

Integrating the ODEs serves two functions for stellar models.

The primary function, as far as the hydrodynamics is concerned, is to provide the magnitude and sign of the energy generation rate. The second function is to describe the evolution of the abundances.

Obtaining accurate values for the energy generation rate is expensive in terms of computer memory and CPU time.

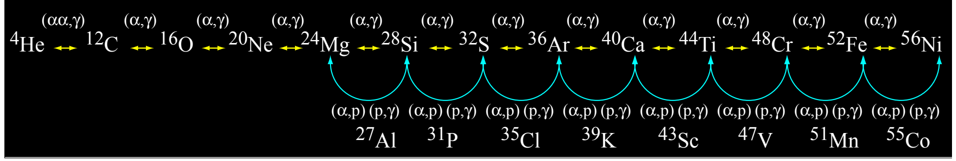
The largest block of memory in a stellar hydrodynamic program is reserved for storing the abundances at every grid point.

For a given set of resources, one must choose between having fewer isotopes or having less spatial resolution.

The general response to this tradeoff has been to evolve fewer isotopes, and thus calculate an approximate energy generation rate.

The set of 13 nuclei most commonly used for this purpose are ^{4}He, ^{12}C, ^{16}O, ^{20}Ne, ^{24}Mg, ^{28}Si, ^{32}S, ^{36}Ar, ^{40}Ca, ^{44}Ti, ^{48}Cr, ^{52}Fe, ^{56}Ni.

Alpha-chain Network



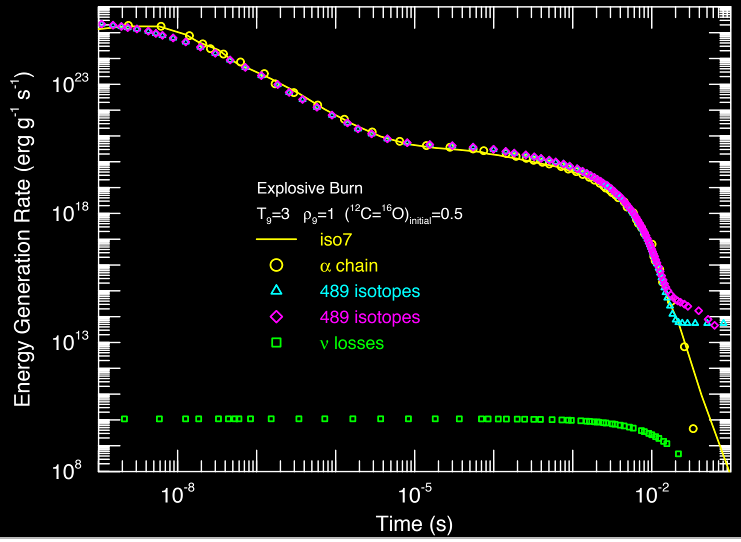
(αα,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ)

^{4}He ↔ ^{12}C ↔ ^{16}O ↔ ^{20}Ne ↔ ^{24}Mg ↔ ^{28}Si ↔ ^{32}S ↔ ^{36}Ar ↔ ^{40}Ca ↔ ^{44}Ti ↔ ^{48}Cr ↔ ^{52}Fe ↔ ^{56}Ni

(α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ)

^{27}Al ^{31}P ^{35}Cl ^{39}K ^{43}Sc ^{47}V ^{51}Mn ^{55}Co

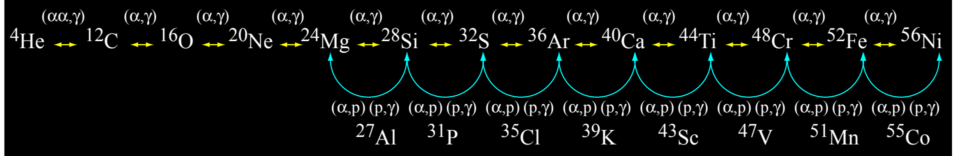
An α-chain network can give a energy generation rate that is generally within 20% of the energy generation rate given by larger reaction networks at a fraction of the cost.



A definition of what we mean by an α-chain network is prudent.

A strict α-chain only has (α,γ) and (γ,α) links among the 13 isotopes ^{4}He, ^{12}C, ^{16}O, ^{20}Ne, ^{24}Mg, ^{28}Si, ^{32}S, ^{36}Ar, ^{40}Ca, ^{44}Ti, ^{48}Cr, ^{52}Fe, and ^{56}Ni.

Alpha-chain Network



(αα,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ)

^{4}He ↔ ^{12}C ↔ ^{16}O ↔ ^{20}Ne ↔ ^{24}Mg ↔ ^{28}Si ↔ ^{32}S ↔ ^{36}Ar ↔ ^{40}Ca ↔ ^{44}Ti ↔ ^{48}Cr ↔ ^{52}Fe ↔ ^{56}Ni

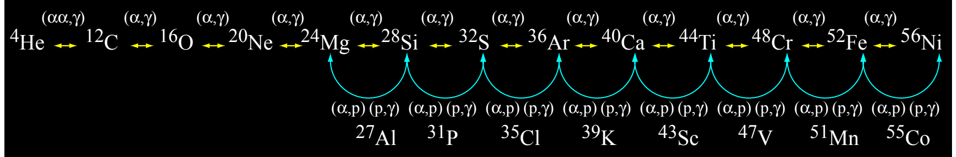
(α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ)

^{27}Al ^{31}P ^{35}Cl ^{39}K ^{43}Sc ^{47}V ^{51}Mn ^{55}Co

Since (α,p)(p,γ) flows are faster than flows through (α,γ) channels, it is essential to include (α,p)(p,γ) and (γ,p)(p,α) links in order to obtain reasonably accurate energy generation rates and abundances when the temperature exceeds ~2.5 x 10^{9} K.

In our α-chain network, we assume steady-state proton flows through the intermediate isotopes ^{27}Al, ^{31}P, ^{35}Cl, ^{39}K, ^{43}Sc, ^{47}V, ^{51}Mn, and ^{55}Co.

Alpha-chain Network



(αα,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ) (α,γ)

^{4}He ↔ ^{12}C ↔ ^{16}O ↔ ^{20}Ne ↔ ^{24}Mg ↔ ^{28}Si ↔ ^{32}S ↔ ^{36}Ar ↔ ^{40}Ca ↔ ^{44}Ti ↔ ^{48}Cr ↔ ^{52}Fe ↔ ^{56}Ni

(α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ) (α,p) (p,γ)

^{27}Al ^{31}P ^{35}Cl ^{39}K ^{43}Sc ^{47}V ^{51}Mn ^{55}Co

This strategy permits inclusion of (α,p)(p,γ) sequences without explicitly evolving the proton or intermediate isotope abundances.

HIPACC Network Projects

- Run the 13 isotope α-chain code in hydrostatic mode for the initial conditions $T = 3x10^9$ K, $\rho = 10^9$ g/cm^3, X(^{4}He) = 1.0.

- Plot the abundance evolution with the supplied gnuplot aprox13.gplt file (or something else).

- What isotope dominates when?

- Run the code in explosive (adiabatic) mode (same initial conditions). Compare and contrast the two results.

- Look at the other reaction network avaliable at http://cococubed.asu.edu/code_pages/burn.shtml