To integrate $du/dt = f(u)$ from $t = t_n$ to $t_{n+1} = t_n + \Delta t_n$, we first apply the trapezoidal rule (TR) to advance the solution from $t_n$ to $t_n + \gamma \Delta t_n$:

$$u_{n+\gamma} - \gamma \frac{\Delta t_n}{2} f_{n+\gamma} = u_n + \gamma \frac{\Delta t_n}{2} f_n,$$

and then use the second-order backward differentiation formula (BDF2)\(^1\) to advance the solution from $t_{n+\gamma}$ to $t_{n+1}$:

$$u_{n+1} - \frac{1 - \gamma}{2 - \gamma} \Delta t_n f_{n+1} = \frac{1}{\gamma(2-\gamma)} u_{n+\gamma} - \frac{(1 - \gamma)^2}{\gamma(2-\gamma)} u_n.$$

This composite one-step method is second-order accurate and L-stable.

We linearize $f_{n+1}$ in Eq. (2) (and similarly $f_{n+\gamma}$ in Eq. (1)) iteratively by setting the new iterative solution to

$$u_{n+1}^{(k+1)} = u_{n+1}^{(k)} + \delta u_{n+1}^{(k)}, \quad u_{n+1}^{(0)} = u_{n+\gamma}$$

and approximating

$$f_{n+1}^{(k+1)} = f_{n+1}^{(k)} + \left( \frac{\partial f}{\partial u} \right)_{n+1}^{(k)} \delta u_{n+1}^{(k)}$$

\(^1\)For BDF2, $e_t = -\frac{(1-\gamma)^2}{\gamma(2-\gamma)} \Delta t^3 u'''$. 

---

TRBDF2


where \( k = 0, 1, \ldots \) labels the Newton iterations. At each TR or BDF2 partial step, we iterate until the Newton method converges.

The Newton equation for the TR partial step is

\[
\begin{pmatrix}
I - \gamma \frac{\Delta t_n}{2} \left( \frac{\partial f}{\partial u} \right)_{n+\gamma}^{(k)}
\end{pmatrix} \delta u_{n+\gamma}^{(k)} = - (u_{n+\gamma}^{(k)} - u_n) + \gamma \frac{\Delta t_n}{2} (f_{n+\gamma}^{(k)} + f_n) \equiv -R_{TR}
\]

where \( R_{TR} \) is the residual for Eq. (1).

The Newton equation for the BDF2 partial step is

\[
\begin{pmatrix}
I - \frac{1 - \gamma}{2 - \gamma} \Delta t_n \left( \frac{\partial f}{\partial u} \right)_{n+1}^{(k)}
\end{pmatrix} \delta u_{n+1}^{(k)} =
- \left( u_{n+1}^{(k)} - \frac{1}{\gamma(2 - \gamma)} u_{n+\gamma} + \frac{(1 - \gamma)^2}{\gamma(2 - \gamma)} u_{n} \right) + \frac{1 - \gamma}{2 - \gamma} \Delta t_n f_{n+1}^{(k)} \equiv -R_{BDF2}
\]

where \( R_{BDF2} \) is the residual for Eq. (2).

The timestep size \( \Delta t \) is adjusted dynamically within a window \([\Delta t_{\text{min}}, \Delta t_{\text{max}}]\) by monitoring a divided-difference estimate of the local error \( e_l \):

\[
e_l = k_{\gamma} \Delta t_n^3 u''''
\]

\[
\approx 2k_{\gamma} \Delta t_n \left( \frac{1}{\gamma} f_n - \frac{1}{\gamma(1 - \gamma)} f_{n+\gamma} + \frac{1}{1 - \gamma} f_{n+1} \right),
\]

where

\[
k_{\gamma} = \frac{-3\gamma^2 + 4\gamma - 2}{12(2 - \gamma)}.
\]

The three values of \( f \) employed in Eq. (8) have already been calculated in the most recent TRBDF2 timestep.

The \( ||e_l|| \) is minimized for \( \gamma = 2 - \sqrt{2} \).

**A-stability and L-stability**

A time integration method for \( du/dt = au \) (Re\{a\} < 0) is **A-stable** if

\[
||u_{n+1}|| \leq ||u_n||.
\]

2
A time integration method for $du/dt = au$ ($\text{Re}\{a\} < 0$) is **L-stable** if it is A-stable and

$$\lim_{\Delta t \to \infty} \frac{||u_{n+1}||}{||u_n||} = 0.$$  \hspace{1cm} (11)

TR is A-stable, but not L-stable. Backward Euler (first- and second-order) and TRBDF2 are L-stable.

Figure 1: Simulation of nonlinear diffusion in semiconductor processing with a large $\Delta t$. 
Figure 2: $G_{TRBDF2}$ as a function of $|a|\Delta t$ for $\gamma = 2 - \sqrt{2}$.

Dynamic Timestep

To dynamically adjust $\Delta t$, we monitor an estimate of the local error $e_l$. Define

$$r = \frac{||e_l||}{\epsilon_R ||u_n|| + \epsilon_A}$$

where $\epsilon_R$ specifies the relative error tolerance and $\epsilon_A$ specifies an absolute error tolerance (“effective zero”) so that the denominator of $r$ never equals zero. In other words, we are requiring

$$||e_l|| \lesssim \epsilon_R ||u_n|| + \epsilon_A.$$

Our dynamic timestep algorithm is

- If $r \leq 2$, accept the provisional solution as $u_{n+1}$ and set the new $\Delta t \leftarrow \Delta t / r^{1/(p+1)}$.
- If $r > 2$, redo the timestep with $\Delta t \leftarrow \Delta t / r^{1/(p+1)}$ or $\Delta t \leftarrow \Delta t / 2$.

The scaling of $\Delta t$ maintains $||e_l|| \approx \epsilon_R ||u_n||$ since we are extrapolating that

$$r_{\text{new}} = \frac{||e_{l,\text{new}}||}{\epsilon_R ||u_{n+1}|| + \epsilon_A} \approx \frac{||e_{l,\text{new}}||}{||e_l||} \approx \frac{\Delta t_{\text{new}}^{p+1}}{\Delta t^{p+1}} r = 1$$

for $\Delta t_{\text{new}} = \Delta t / r^{1/(p+1)}$. 
