Final Report-
SPECTRAL DEFERRED CORRECTION METHODS FOR ORDINARY DIFFERENTIAL EQUATIONS

314652013 陳勇捷

1 Introduction and Problem Description

1.1 Problem Setting and Basic Assumptions

We consider the numerical solution of initial value problems for ODEs of the form

$$\begin{cases} \varphi'(t) = F(t, \varphi(t)), \quad t \in [a, b] \\ \varphi(a) = \phi_a, \end{cases}$$

where $\varphi(t) \in \mathbf{C}^n$ and $F : \mathbf{R} \times \mathbf{C} \to \mathbf{C}^n$. Throughout this report, the function F is assumed to be sufficiently smooth, so that the IVP admits a locally unique solution on the interval.

1.2 Stiff v.s. Non-stiff Problems

The classification is determined by the stiffness ratio S derived from the linear test equation

$$\varphi'(t) = \lambda\varphi(t), \quad \Re(\lambda) \leq 0$$

The magnitude of $\Re(\lambda)$ reflects the time scale of decay and plays a central role in determining the numerical behavior of time-stepping schemes.

1.2.1 Non-stiff problems

Definition : A system is **non-stiff** when its time scales are comparable, implying a moderate stiffness ratio:

$$S = \frac{max|\Re(\lambda)|}{min|\Re(\lambda)|} \approx O(1)$$

- **Constraint**: The step size h is primarily determined by **accuracy requirements**. Stability constraints are mild.
- **Strategy**: **Explicit methods** (e.g., explicit Runge-Kutta) are preferred due to low computational cost per step.

1.2.2 Stiff problems

Definition : A system is **stiff** when it contains widely separated time scales ($S \gg 1$), typically characterized by eigenvalues with large negative real parts ($\Re(\lambda) \ll 0$).

- **Dynamics** : The solution contains fast-decaying transients.
- **Constraint**: **Stability restrictions dominate**. For explicit methods, h must be impractically small to satisfy stability regions ($|1 + h\lambda| < 1$).
- **Strategy**: **Implicit methods** (e.g., BDF, Implicit RK) are required. Their A-stability properties allow larger time steps determined solely by the active dynamics.

1.3 Stability concepts: A-stability and L-stability

The suitability of a numerical method for stiff problems is commonly assessed through its behavior on the linear test equation $y' = \lambda y$ with $\Re(\lambda) \leq 0$. Applying a one-step method yields the relation:

$$y_{n+1} = A(z)y_n, \quad z = h\lambda$$

where $A(z)$ is the amplification factor of the method.

1.3.1 A-Stability (Unconditional Stability)

Definition: A method is **A-stable** if its stability region covers the entire left half-plane:

$$|A(z)| \leq 1 \quad \text{for all } \Re(z) \leq 0$$

Implication: This is a minimal requirement for stiff solvers. It ensures the numerical solution remains bounded for decaying inputs, removing stability restrictions on h so the step size is determined solely by accuracy.

1.3.2 L-Stability (Asymptotic Damping)

Some A-stable methods (e.g., Trapezoidal Rule) have $|A(z)| \to 1$ as $\Re(z) \to -\infty$, causing spurious oscillations for very stiff modes.

Definition: A method is **L-stable** if it is A-stable *and* possesses strong asymptotic damping:

$$\lim_{\Re(z)\to-\infty} A(z) = 0$$

Implication: L-stability ensures that rapidly decaying components ($\lambda \ll 0$) are immediately damped out in the numerical solution, preventing the ”ringing” artifacts often seen in simple A-stable schemes.

2 Deferred Correction Methods

This chapter reviews the fundamental principles of Classical Deferred Correction(CDC), analyzes its limitations regarding numerical stability, and details the Spectral Deferred Correction(SDC) method proposed in this study. SDC overcomes the intrinsic defects of traditional methods related to numerical differentiation and grid selection by reformulating the problem via the Picard integral equation and employing spectral integration techniques.

2.1 Classical Deferred Correction (CDC)

The core concept of deferred correction is to utilize low-order numerical methods to iteratively solve an error equation, thereby progressively increasing the order of accuracy of the numerical solution. Consider the IVP on the interval $[a, b]$:

$$\varphi'(t) = F(t, \varphi(t)), \quad \varphi(a) = \varphi_a$$

We define a grid with $m + 1$ equispaced nodes $t_i = a + i \cdot h$ ($i = 0, \dots, m$), where h is the step size. Suppose we first compute an approximate solution $\eta = (\eta_1, \dots, \eta_m)$ using a method of order k, such that the error satisfies $\eta_i = \varphi(t_i) + O(h^k)$. We then construct the interpolating polynomial $L^m(\eta, t)$ passing through these approximate values.

The error function $\delta(t)$ is defined as the difference between the exact solution and the interpolant :

$$\delta(t) = \varphi(t) - L^m(\eta, t)$$

Differentiating this expression yields the differential equation satisfied by $\delta(t)$:

$$\delta'(t) = F(t, \delta(t) + L^m(\eta, t)) - \frac{d}{dt}L^m(\eta, t), \quad \delta(0) = 0$$

By solving this error equation numerically, we obtain a correction term that increases the order of accuracy of the solution .

However, the practical application of this classical approach is limited by two well-known numerical instabilities. First, the use of equispaced nodes for high-degree interpolation leads to the Runge phenomenon, causing severe oscillations near interval boundaries. Second, the formulation requires numerical differentiation ($\frac{d}{dt}L^m$), an ill-conditioned operation that amplifies numerical noise as the order increases. These factors prevent the classical method from being effectively extended to very high orders.

2.2 Spectral Deferred Correction (SDC)

To overcome the limitations of equispaced grids and numerical differentiation, the SDC method reformulates the problem using integral equations and employs spectral methods on Gaussian grids.

2.2.1 The Picard Integral Formulation

Instead of relying on the differential form, SDC utilizes the Picard integral equation :

$$\varphi(t) = \varphi_a + \int_a^t F(\tau, \varphi(\tau))d\tau$$

For a current approximation $\varphi^0(t)$, the residual $\epsilon(t)$ is defined as the difference between the integrated right-hand side and the approximation. Substituting the error $\delta(t) = \varphi(t) - \varphi^0(t)$ into the Picard equation yields the integral error equation :

$$\delta(t) - \int_a^t [F(s, \varphi^0(s) + \delta(s)) - F(s, \varphi^0(s))]ds = \epsilon(t)$$

This formulation avoids the derivative term $\frac{d}{dt}L^m$ entirely. Since integration is a smoothing operator, this approach eliminates the noise amplification associated with numerical differentiation in the classical method.

2.2.2 Spectral Integration with Gaussian Nodes

To ensure stability for high-order approximations, SDC replaces equispaced nodes with Gauss-Legendre nodes $s_1, \ldots, s_m$.

We define the Spectral Integration Matrix S^m. If f is a vector of function values at the Gauss nodes, the integral vector $g = S^m f$ represents the exact integral of the Lagrange interpolant $L^m(f, t)$:

$$g_i = \int_{-1}^{t_i} L^m(f, t)dt$$

Unlike differentiation matrices, the integration matrix S^m is well-conditioned, with a bounded maximum eigenvalue and a minimum eigenvalue of order $O(1/m^2)$. This property allows the order of the method to be increased significantly without inducing numerical instability.

2.2.3 The Iterative Correction Scheme

The SDC algorithm combines these concepts into a robust iterative procedure :

1. **Initialization** : Compute an initial solution $\varphi^{[0]}$ on Gauss nodes using a low-order Euler method (Forward for non-stiff, Backward for stiff).

2. **Successive Corrections** : For iteration $j = 1, \ldots, J$:

- Compute the residual $\epsilon(t)$ using the spectral integration matrix S^m.
- Solve the integral error equation using the same Euler method to obtain $\delta^{[j]}$.
- Update the solution: $\varphi^{[j]} = \varphi^{[j-1]} + \delta^{[j]}$.

This process progressively improves the accuracy, converging to order $O(h^{\min(m,J+1)})$, while maintaining the stability characteristics of the underlying Euler solver.

3 Stability Analysis and Findings

This chapter analyzes the stability of SDC methods for stiff problems, highlighting the limitations of the basic implicit scheme and the novel strategy used to achieve high-order L-stability.

3.1 Stability Limits of the $EuImp$ Scheme

The basic SDC scheme for stiff problems ($EuImp$), driven by Implicit Euler, is A-stable but fails to be L-stable.

The amplification factor $Am(\lambda)$ of $EuImp$ does not vanish at infinity. Instead, it converges to a non-zero constant $\mu(m, J)$:

$$\lim_{|\lambda|\to\infty} Am(\lambda) = \mu(m, J) \neq 0$$

This implies that extremely stiff components (Re$(\lambda) \ll 0$) are not fully damped, leading to potential residual oscillations.

3.2 The Linear Combination Strategy

To enforce L-stability, the authors introduce a linear combination strategy ($EuComb$).

The core idea is combining two $EuImp$ schemes with different parameters, the non-zero limit μ is cancelled out :

$$EuComb = \frac{\mu_2 \cdot EuImp^{(1)} - \mu_1 \cdot EuImp^{(2)}}{\mu_2 - \mu_1} \quad \Rightarrow \quad \lim_{|\lambda|\to\infty} Am(\lambda) = 0$$

Then $EuComb$ is L-stable.

Key Findings:

- **Arbitrary Order** : Successfully constructed L-stable schemes up to order 19.
- **Optimal Stability** : High-order schemes (e.g., $EuComb^{19,19}_{20,19}$) achieve stability angles $\alpha > 89.99°$, effectively behaving as A-stable solvers.

4 Implementation and Numerical Validation

In this section, we empirically validate the proposed SDC framework using a fully adaptive ODE solver. We verify the theoretical claims regarding accuracy and stability by benchmarking against the Robertson chemical kinetics problem, a classic stiff test case.

4.1 Experimental Setup

4.1.1 The Robertson Problem

$$\begin{aligned} y_1' &= -0.04y_1 + 10^4 y_2 y_3 \\ y_2' &= 0.04y_1 - 10^4 y_2 y_3 - 3 \cdot 10^7 y_2^2 \\ y_3' &= 3 \cdot 10^7 y_2^2 \end{aligned}$$

The simulation is performed on $t \in [0, 1]$ with initial conditions $y(0) = [1, 0, 0]^T$. The problem is characterized by extreme stiffness, as reaction rates vary by orders of magnitude.

4.1.2 Unified Adaptive Controller

To ensure a strictly fair comparison, all methods utilize an identical step-doubling adaptive controller.

- **Mechanism** : For every step, the solution is computed once with step size h and twice with $h/2$.
- **Tolerance** : The local error estimate is checked against $Tol = 10^{-10}$.
- **Fairness** : This ensures that any difference in performance arises solely from the numerical scheme's ability to take large steps accurately, rather than differences in heuristic step-size strategies.

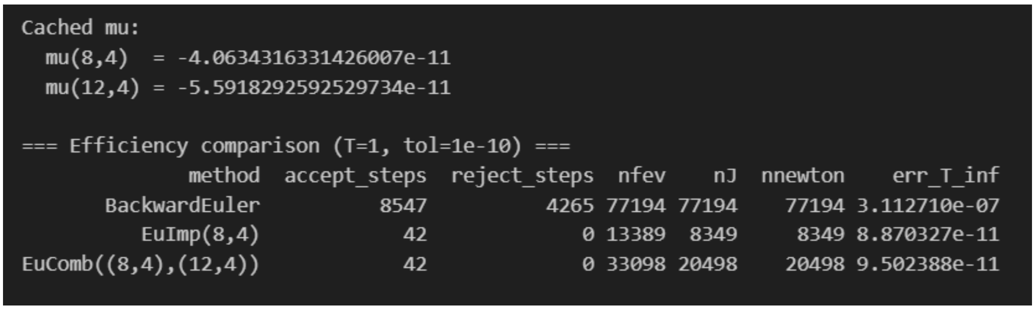
4.2 Methodology Positioning

Based on the algorithm design, we categorize the solvers into three distinct tiers:

1. **Backward Euler (BE)** : The Baseline. A first-order, naturally L-stable method. While robust, its low order ($O(h)$) forces it to take extremely small steps to meet high precision requirements.
2. $EuImp(8, 4)$: The Efficient Engine. An implicit SDC scheme using $m = 8$ Gauss nodes and 4 correction sweeps. It is A-stable and provides high-order accuracy.
3. $EuComb$: The Stability Guarantee. A linear combination of EuImp(8,4) and EuImp(12,4). Its mathematical purpose is to extrapolate the asymptotic amplification factor to zero, theoretically enforcing L-stability.

4.3 Results and Critical Analysis

The performance data for the simulation is summarized in Figure 1.



```
Cached mu:
  mu(8,4)  = -4.0634316331426007e-11
  mu(12,4) = -5.5918292592529734e-11

=== Efficiency comparison (T=1, tol=1e-10) ===
              method  accept_steps  reject_steps  nfev     nJ  nnewton    err_T_inf
       BackwardEuler          8547          4265 77194 77194    77194 3.112710e-07
          EuImp(8,4)            42             0 13389  8349     8349 8.870327e-11
EuComb((8,4),(12,4))            42             0 33098 20498    20498 9.502388e-11
```

Figure 1: result

4.3.1 Step Size and Robustness (Accept/Reject Rates)

The `accept_steps` metric is the most direct indicator of a method's efficiency—fewer steps imply a larger average step size.

- **High-Order Power** : Both $EuImp$ and $EuComb$ completed the integration in only 42 steps, whereas Backward Euler required 8,547 steps. This indicates that the average step size of the SDC methods was approximately 200 times larger than that of the first-order baseline.
- **Adaptive Friction (Reject Steps)** : `reject_steps` represents the ”friction” in the adaptive process.
  - BE (4265 rejects) : The high rejection rate (33%) indicates that the low-order method frequently failed error estimates whenever the step size increased slightly, forcing the controller to halve the step repeatedly.
  - $EuImp$/$EuComb$ (0 rejects) : The zero rejection rate demonstrates extreme robustness. These methods navigated the stiff transients smoothly, satisfying the error tolerance on the first attempt for every step.

4.3.2 Performance Evaluation: Cost vs. Accuracy

In stiff systems, true performance is defined by the trade-off between computational effort (function evaluations/Newton iterations) and the achieved global precision.

- **Backward Euler** : Backward Euler reveals the critical limitation of low-order schemes. Despite the adaptive controller enforcing a local tolerance of 10^{-10}, it accumulated a global error of $\approx 3 \times 10^{-7}$ while requiring over 77,000 function evaluations. This highlights that for low-order methods, local error control implies excessive computational cost without guaranteeing tight global accuracy.

- $EuImp$: $EuImp$ emerged as the most efficient solver. It reduced the computational workload by a factor of $\approx 5.8\times$ compared to Backward Euler (13k vs 77k evaluations) while successfully achieving a global error of 10^{-11}. This represents an order-of-magnitude reduction in Newton iterations and strict adherence to the target tolerance.
- $EuComb$: $EuComb$ matched the superior accuracy of $EuImp$ (10^{-11}) but incurred a $\approx 2.5\times$ higher workload (in terms of `nfev` and `nnewton`). This overhead is the expected price for computing two separate SDC solutions per step to algorithmically enforce L-stability.

5 Summary or conclusion

In conclusion, this study demonstrates that Spectral Deferred Correction (SDC) methods effectively bridge the gap between high-order accuracy and stiff stability. Our numerical validation reveals that while the traditional Backward Euler method remains robust, its inefficiency renders it unsuitable for high-precision applications. In contrast, the $EuImp$ scheme emerges as the optimal solution for general stiff problems, achieving orders-of-magnitude efficiency gains by combining A-stability with high-order spectral corrections. Furthermore, the $EuComb$ strategy serves as a specialized, L-stable alternative, providing a necessary guarantee of asymptotic damping in extreme stiffness regimes, albeit at a manageable additional computational cost.