Section 6.5

Math 231

Hope College

- Euler's Method uses the tangent line to approximate solutions to a first-order IVP of the form

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0.$$

- The approximation is based on a step size Δt and a number of steps, n.
- We set

$$t_{j+1} = t_j + \Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- Each successive y value depends on the previous approximation using the formula

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- Euler's Method uses the tangent line to approximate solutions to a first-order IVP of the form

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0.$$

- The approximation is based on a step size Δt and a number of steps, n.
- We set

$$t_{j+1} = t_j + \Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- Each successive y value depends on the previous approximation using the formula

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

Euler's Method

- Euler's Method uses the tangent line to approximate solutions to a first-order IVP of the form

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0.$$

- The approximation is based on a step size Δt and a number of steps, n.
- We set

$$t_{j+1} = t_j + \Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- Each successive y value depends on the previous approximation using the formula

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- Euler's Method uses the tangent line to approximate solutions to a first-order IVP of the form

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0.$$

- The approximation is based on a step size Δt and a number of steps, n.
- We set

$$t_{j+1} = t_j + \Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- Each successive y value depends on the previous approximation using the formula

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

Errors in Numerical Methods

Assume that we are approximating a function value $f(t_n)$ by y_n resulting from an iterative process $(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$. There are 4 main types of error:

- **Overall error:** $f(t_n) - y_n$. This is usually impossible to compute precisely.
- **Round-off error:** The error caused by doing numerical computations inside a computer or calculator.
- **Local error:** The error LE_j committed in the approximation used at the jth step of the procedure. This does not include any round-off error committed, but only the error caused by the type of approximation itself.
- **Global error:** The part of the overall error not due to roundoff error. Global error comes from two places: the accumulation of local error from each step in the process and the fact that the process at step j is using only an approximate initial condition.

Errors in Numerical Methods

Assume that we are approximating a function value $f(t_n)$ by y_n resulting from an iterative process $(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$. There are 4 main types of error:

- **Overall error:** $f(t_n) - y_n$. This is usually impossible to compute precisely.
- **Round-off error:** The error caused by doing numerical computations inside a computer or calculator.
- **Local error:** The error LE_j committed in the approximation used at the jth step of the procedure. This does not include any round-off error committed, but only the error caused by the type of approximation itself.
- **Global error:** The part of the overall error not due to roundoff error. Global error comes from two places: the accumulation of local error from each step in the process and the fact that the process at step j is using only an approximate initial condition.

Errors in Numerical Methods

Assume that we are approximating a function value $f(t_n)$ by y_n resulting from an iterative process $(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$. There are 4 main types of error:

- **Overall error:** $f(t_n) - y_n$. This is usually impossible to compute precisely.
- **Round-off error:** The error caused by doing numerical computations inside a computer or calculator.
- **Local error:** The error LE_j committed in the approximation used at the jth step of the procedure. This does not include any round-off error committed, but only the error caused by the type of approximation itself.
- **Global error:** The part of the overall error not due to roundoff error. Global error comes from two places: the accumulation of local error from each step in the process and the fact that the process at step j is using only an approximate initial condition.

Errors in Numerical Methods

Assume that we are approximating a function value $f(t_n)$ by y_n resulting from an iterative process $(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$. There are 4 main types of error:

- **Overall error:** $f(t_n) - y_n$. This is usually impossible to compute precisely.
- **Round-off error:** The error caused by doing numerical computations inside a computer or calculator.
- **Local error:** The error LE_j committed in the approximation used at the jth step of the procedure. This does not include any round-off error committed, but only the error caused by the type of approximation itself.
- **Global error:** The part of the overall error not due to roundoff error. Global error comes from two places: the accumulation of local error from each step in the process and the fact that the process at step j is using only an approximate initial condition.

Errors in Euler's Method

Theorem 6.32: Suppose Euler's Method is being run on an IVP of the form

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0$$

with n steps to approximate the value $y(t_n)$. If the solution curve $y = f(t)$ to this IVP has a continuous second derivative on an open interval containing $[t_0, t_n]$, then

1. the local error at each stage is at most proportional to $1/n^2$.
2. the global error is at most proportional to $1/n$.

Taylor Series Methods

- Euler's Method can be extended to a Taylor series method for approximating solutions to

$$y' = g(t, y) \qquad y(t_0) = y_0.$$

- As before,

$$t_{j+1} = t_j + \Delta t \qquad \text{for } j = 0, 1, \ldots, n-1.$$

- In a kth order Taylor series method, the iterative formula is replaced by

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t + \frac{g'(t_j, y_j)}{2!}(\Delta t)^2 + \cdots + \frac{g^{(k-1)}(t_j, y_j)}{k!}(\Delta t)^k.$$

- The local error in the kth order Taylor series method is proportional to $1/n^{k+1}$, while the global error is proportional to $1/n^k$. (Theorem 6.35)

Taylor Series Methods

- Euler's Method can be extended to a Taylor series method for approximating solutions to

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0.$$

- As before,

$$t_{j+1} = t_j + \Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- In a kth order Taylor series method, the iterative formula is replaced by

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t + \frac{g'(t_j, y_j)}{2!}(\Delta t)^2 + \cdots + \frac{g^{(k-1)}(t_j, y_j)}{k!}(\Delta t)^k.$$

- The local error in the kth order Taylor series method is proportional to $1/n^{k+1}$, while the global error is proportional to $1/n^k$. (Theorem 6.35)

Taylor Series Methods

- Euler's Method can be extended to a Taylor series method for approximating solutions to

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0.$$

- As before,

$$t_{j+1} = t_j + \Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- In a kth order Taylor series method, the iterative formula is replaced by

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t + \frac{g'(t_j, y_j)}{2!}(\Delta t)^2 + \cdots + \frac{g^{(k-1)}(t_j, y_j)}{k!}(\Delta t)^k.$$

- The local error in the kth order Taylor series method is proportional to $1/n^{k+1}$, while the global error is proportional to $1/n^k$. (Theorem 6.35)

Taylor Series Methods

- Euler's Method can be extended to a Taylor series method for approximating solutions to

$$y' = g(t, y) \qquad\qquad y(t_0) = y_0.$$

- As before,

$$t_{j+1} = t_j + \Delta t \qquad\qquad \text{for } j = 0, 1, \ldots, n-1.$$

- In a kth order Taylor series method, the iterative formula is replaced by

$$y_{j+1} = y_j + g(t_j, y_j)\Delta t + \frac{g'(t_j, y_j)}{2!}(\Delta t)^2 + \cdots + \frac{g^{(k-1)}(t_j, y_j)}{k!}(\Delta t)^k.$$

- The local error in the kth order Taylor series method is proportional to $1/n^{k+1}$, while the global error is proportional to $1/n^k$. (Theorem 6.35)

The Runge-Kutta Method

- The Runge-Kutta Method is another iterative method that is easier to apply than the higher-order Taylor methods, yet still gives good error behavior.
- The local error in the Runge-Kutta method is proportional to $1/n^5$, while the global error is proportional to $1/n^4$.
- Sage code for the Runge-Kutta method is in the book.

The Runge-Kutta Method

- The Runge-Kutta Method is another iterative method that is easier to apply than the higher-order Taylor methods, yet still gives good error behavior.
- The local error in the Runge-Kutta method is proportional to $1/n^5$, while the global error is proportional to $1/n^4$.
- Sage code for the Runge-Kutta method is in the book.

The Runge-Kutta Method

- The Runge-Kutta Method is another iterative method that is easier to apply than the higher-order Taylor methods, yet still gives good error behavior.
- The local error in the Runge-Kutta method is proportional to $1/n^5$, while the global error is proportional to $1/n^4$.
- Sage code for the Runge-Kutta method is in the book.