**Study Guide: Numerical Methods for Solving ODEs**

**1. Overview of Numerical Methods for ODEs**

Ordinary Differential Equations (ODEs) describe how a function changes over time. Analytical solutions are not always possible, so **numerical methods** provide approximate solutions.

Numerical methods for ODEs are mainly categorized as:

1. **Single-Step Methods** – Use only the current value to compute the next step.
2. **Multi-Step Methods** – Use previous values to estimate future values.
3. **Predictor-Corrector Methods** – Predict a value, then refine it.

**2. Classification of Methods**

**Single-Step Methods**

| **Method** | **Description** | **Advantages** | **Disadvantages** |
| --- | --- | --- | --- |
| **Taylor’s Method** | Uses Taylor series expansion to approximate the function. | Very accurate if derivatives are known. | Requires higher-order derivatives. |
| **Euler’s Method** | Uses the slope at the current point to estimate the next value. | Simple and easy to implement. | Low accuracy, large errors. |
| **Modified Euler’s Method** | Uses an improved slope by averaging two slopes. | More accurate than Euler’s method. | More computations required. |
| **Runge-Kutta (RK4)** | Uses multiple weighted slopes for better accuracy. | Highly accurate, widely used. | More computationally intensive. |

**Multi-Step Methods**

| **Method** | **Description** | **Advantages** | **Disadvantages** |
| --- | --- | --- | --- |
| **Adams-Bashforth** | Explicit method that uses past values to predict the next value. | Efficient for large steps. | Less stable than implicit methods. |
| **Adams-Moulton** | Implicit method that corrects predictions from Adams-Bashforth. | More stable and accurate. | Requires solving implicit equations. |
| **Milne’s Method** | Uses Simpson’s rule for predicting and trapezoidal correction. | Good for smooth functions. | Needs multiple past values. |

**Predictor-Corrector Methods**

* **Adams-Bashforth & Adams-Moulton** (Predictor: Explicit, Corrector: Implicit)
* **Milne’s Method** (Predictor: Simpson’s Rule, Corrector: Trapezoidal Rule)

**3. Key Concepts**

* **Numerical solutions for ODEs are called** → *Initial Value Problems (IVP).*
* **Method that needs higher derivatives** → *Taylor’s Method.*
* **First category of numerical methods for ODEs** → *Taylor’s Method.*
* **Most widely used method in practice** → *Runge-Kutta (RK4).*
* **Common Predictor-Corrector Pair** → *Adams-Bashforth & Adams-Moulton.*
* **Multi-step method using Simpson’s rule for prediction** → *Milne’s Method.*

**4. Comparison Table**

| **Method** | **Type** | **Explicit / Implicit** | **Requires Multiple Previous Steps?** |
| --- | --- | --- | --- |
| Taylor’s Method | Single-Step | Explicit | No |
| Euler’s Method | Single-Step | Explicit | No |
| Modified Euler | Single-Step | Explicit | No |
| Runge-Kutta (RK4) | Single-Step | Explicit | No |
| Adams-Bashforth | Multi-Step | Explicit | Yes |
| Adams-Moulton | Multi-Step | Implicit | Yes |
| Milne’s Method | Multi-Step | Predictor-Corrector | Yes |

**5. Sample Questions**

1. **Which numerical method requires prior calculations of higher derivatives?**  
   **Answer:** Taylor’s Method
2. **What is the first category of numerical methods for solving ODEs?**  
   **Answer:** Taylor’s Method
3. **Which method is a widely used practical alternative to Taylor’s Method?**  
   **Answer:** Runge-Kutta (RK4)
4. **What is a numerical solution for an ODE called?**  
   **Answer:** Initial Value Problem (IVP)
5. **Which method uses Simpson’s Rule for prediction and the Trapezoidal Rule for correction?**  
   **Answer:** Milne’s Method
6. **Which methods fall under the Predictor-Corrector category?**  
   **Answer:** Adams-Bashforth & Adams-Moulton, Milne’s Method
7. **Which method is the simplest explicit method for solving ODEs?**  
   **Answer:** Euler’s Method
8. **What is the main disadvantage of Euler’s Method?**  
   **Answer:** It has large errors and low accuracy.

**Understanding the Order of Error in Numerical Methods for ODEs**

The **order of error** in numerical methods refers to how the **approximation error** behaves as the step size h decreases. It gives an indication of **accuracy and convergence** of the method.

**1. Order of Error in Different Methods**

| **Method** | **Local Error (per step)** | **Global Error (overall solution)** |
| --- | --- | --- |
| **Euler’s Method** | O(h^2) | O(h) |
| **Modified Euler’s Method** | O(h^3) | O(h^2) |
| **Runge-Kutta (RK4)** | O(h^5) | O(h^4) |
| **Adams-Bashforth** | Depends on order | Varies |
| **Adams-Moulton** | Depends on order | Varies |
| **Milne’s Method** | O(h^5) | O(h^4) |

* **Local Error** → Error per **single** step.
* **Global Error** → Error **accumulated** over multiple steps.

**Key Takeaways:**

* **Higher-order methods (RK4, Adams, Milne) have smaller error terms.**
* **Euler’s method has the highest error and is less accurate.**
* **Global error is usually one order lower than local error.**
* **Reducing hhh improves accuracy, but at higher computational cost.**

**2. Additional Points to Add to ODE Notes**

Now that you’ve spotted the **order of error**, here are other **important points** that can be added to your notes:

**2.1 Stability of Numerical Methods**

* Some methods remain stable only if **step size hhh is small**.
* **Implicit methods (Adams-Moulton, Backward Euler)** are more stable than **explicit methods (Euler, RK4).**
* Stability is crucial for **stiff ODEs** (where solutions change rapidly).

**2.2 Convergence**

* A method is **convergent** if its solution **approaches the true solution** as h→0h \to 0h→0.
* Runge-Kutta methods **converge faster** than Euler’s method.

**2.3 Computational Efficiency**

* Higher-order methods (RK4, Milne) **reduce error but require more function evaluations**.
* Trade-off between **accuracy and computational cost**.

**2.4 Predictor-Corrector Methods**

* **Example:** Adams-Bashforth (Predictor) + Adams-Moulton (Corrector).
* Predictor estimates, Corrector refines for better accuracy.

**2.5 Choosing the Right Method**

* **Euler’s Method** → **Fast but inaccurate**; used for rough estimates.
* **Modified Euler / RK2** → **Better accuracy than Euler, still simple**.
* **RK4** → **Balanced accuracy and efficiency** (widely used).
* **Adams & Milne** → **Good for multi-step problems**.

**6. Additional Notes**

* **Runge-Kutta methods** provide a practical alternative to Taylor’s Method without needing higher derivatives.
* **Multi-step methods** require previous values but can achieve high accuracy with fewer function evaluations.
* **Predictor-Corrector methods** improve accuracy by refining predictions.

**Picard’s Method** is an **iterative** method used for solving **initial value problems (IVPs)** of **ordinary differential equations (ODEs)**. It is based on **successive approximations** and is classified as an **analytical method**, not a numerical stepwise method like Euler or Runge-Kutta.

**1. Type of Method:**

✅ **Analytical (Successive Approximation Method)**  
✅ **Integral Equation-Based**  
✅ **Iterative Convergence Approach**

Unlike Euler’s method, which progresses in small steps, **Picard’s method improves the solution iteratively by refining an integral form of the equation**.

**2. Picard’s Method Formulation**

For a first-order ODE:

y′=f(x,y),y(x0)=y0y' = f(x, y), \quad y(x\_0) = y\_0y′=f(x,y),y(x0​)=y0​

Picard’s method converts this into an **integral equation**:

y(x)=y0+∫x0xf(t,y(t))dty(x) = y\_0 + \int\_{x\_0}^{x} f(t, y(t)) dty(x)=y0​+∫x0​x​f(t,y(t))dt

Instead of approximating derivatives (like Euler), **Picard refines y(x)y(x)y(x) iteratively** using **successive approximations**.

**3. Is Picard a Numerical or Analytical Method?**

🔹 **It is not purely numerical like Euler or Runge-Kutta.**  
🔹 It is **semi-numerical** because it **generates a sequence of better approximations.**  
🔹 If continued indefinitely, it **converges to the exact solution**, unlike Euler which gives an approximation based on step size.  
🔹 It is more of a **theoretical method** and is used to **prove the existence and uniqueness of solutions**.

**4. Why Isn’t Picard Used in Practice?**

* **Slow Convergence**: Requires multiple iterations to refine the integral equation.
* **Computationally Expensive**: In real-world applications, Euler or RK4 are preferred for efficiency.
* **Used More for Theoretical Understanding**: Helps prove that solutions to ODEs exist.

**5. Summary Table:**

| **Method** | **Type** | **Approach** | **Accuracy** | **Common Use** |
| --- | --- | --- | --- | --- |
| **Euler’s Method** | Numerical | Stepwise Approximation | O(h)O(h)O(h) | Fast but less accurate |
| **Runge-Kutta (RK4)** | Numerical | Weighted Slopes | O(h4)O(h^4)O(h4) | High accuracy, widely used |
| **Picard’s Method** | Analytical | Integral Iteration | Exact (if continued) | Theoretical proofs, not practical |

**Conclusion**

**Picard’s Method is an analytical iterative method based on integral equations, not a numerical stepwise method like Euler or RK.**

**Picard’s Method** is different from Taylor series because:

1. **No summation of terms like in Taylor Series**
   * Taylor’s method expands the function as an **infinite sum** of derivatives evaluated at a point.
   * Picard’s method **iteratively refines an integral equation** instead of summing up derivatives.
2. **Successive Iteration Instead of Stepwise Approximation**
   * Each iteration of Picard’s method produces a **better approximation** of the solution.
   * It **does not use step size hh explicitly**, unlike Euler or RK4.
3. **Why is y0y\_0 Present in All Iterations?**
   * The formula keeps the **initial value y0y\_0 in every iteration** because the integral equation always builds upon the previous result.
   * Every iteration **refines the function while keeping the initial condition intact**.

### **Comparison of Taylor vs. Picard**

| **Method** | **How it Works** | **Form of Approximation** | **Improvement Over Iterations** |
| --- | --- | --- | --- |
| **Taylor Series** | Expands as power series | y(x)=y(0)+xy′(0)+x22!y′′(0)+…y(x) = y(0) + x y'(0) + \frac{x^2}{2!} y''(0) + \dots | More terms improve accuracy |
| **Picard’s Method** | Iterates integral equation | yn+1(x)=y0+∫x0xf(t,yn(t))dty\_{n+1}(x) = y\_0 + \int\_{x\_0}^{x} f(t, y\_n(t)) dt | Refines entire function per iteration |

### **Conclusion:**

✔ **Taylor: Summation of terms** (like a polynomial expansion).  
✔ **Picard: Successive refinement using integration**, keeping y0y\_0 in all iterations.  
✔ **Both methods improve accuracy, but Picard is more of a theoretical tool, while Taylor is computationally practical.**

**Picard’s Method is not considered a multi-step method**, even though it refines the solution iteratively. The reason lies in **how it generates the next approximation** compared to traditional **multi-step numerical methods**.

### **1. What Defines a Multi-Step Method?**

A **multi-step method** (like Adams-Bashforth, Adams-Moulton, or Milne’s Method) uses **previous values** of yy and y′y' to compute the next step.

**General form of a multi-step method:**

yn+1=f(yn,yn−1,yn−2,… )y\_{n+1} = f(y\_n, y\_{n-1}, y\_{n-2}, \dots)

✅ **Key Feature:** Uses values from multiple past steps to calculate the next one.

### **2. Why Isn’t Picard a Multi-Step Method?**

Picard’s Method:

1. **Does not use past computed values explicitly**.
   * Each iteration **recomputes the entire function** from the integral form, rather than stepping forward using previous computed values.
2. **Each iteration refines the entire function, not just one step.**
   * Unlike Euler or Adams methods, which calculate yn+1y\_{n+1} from yny\_n, Picard generates an **entirely new function** at each iteration.
3. **No step size hh is involved.**
   * Multi-step methods require a **fixed step size hh** to approximate values.
   * Picard’s method works with an **integral equation**, not discrete steps.

### **3. Summary Table: Picard vs. Multi-Step Methods**

| **Feature** | **Picard’s Method** | **Multi-Step Methods (e.g., Adams-Moulton)** |
| --- | --- | --- |
| **Uses past values?** | ❌ No | ✅ Yes |
| **Step-based computation?** | ❌ No | ✅ Yes |
| **Uses explicit step size hh?** | ❌ No | ✅ Yes |
| **Refines entire function?** | ✅ Yes | ❌ No (just next value) |
| **Analytical or Numerical?** | Mostly Analytical | Fully Numerical |

### **Conclusion:**

**Picard is an iterative integral method, not a multi-step method, because it does not rely on past step values but rather refines the entire function each iteration.**

**Picard’s Method:**

* **No step size hh** → Unlike Euler or RK methods, Picard does not depend on step-by-step calculations.
* **No explicit substitution of past values** → It refines the function iteratively using integration, rather than computing new points based on previous ones.

Instead, **it builds an integral equation and improves it over successive iterations.** This is why it’s an **analytical method rather than a numerical multi-step method**.

### **Key Differences from Step-Based Methods**

| **Feature** | **Picard’s Method** | **Euler / RK / Multi-Step Methods** |
| --- | --- | --- |
| Uses step size hh? | ❌ No | ✅ Yes |
| Uses previous values? | ❌ No | ✅ Yes |
| Works via function refinement? | ✅ Yes | ❌ No |
| Uses explicit numerical approximations? | ❌ No (purely analytical) | ✅ Yes |
| Solves by integration? | ✅ Yes | ❌ No (uses finite differences) |

### **Key Takeaway**

**Picard is based on function iteration rather than numerical step-by-step substitution, making it different from Euler and multi-step methods.**