Runge-Kutta Methods with Rooted Trees

Final Presentation

Author: Richard Douglas *Supervisor:* Dr. Shengda Hu

MA489 Research Seminar at Wilfrid Laurier University

March 13, 2014

Outline

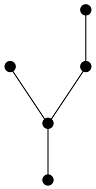
Rooted Trees Definition

Definition

A **rooted tree** is a connected graph with directed edges and a finite number of vertices such that

- there is one vertex designated as the **root**,
- every vertex except the root has exactly one parent, that is, if a vertex is not the root then there is exactly one edge directed at it.

Example:

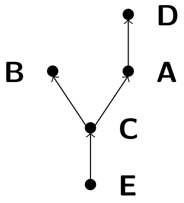


Vertices that are not the parent of any other vertices are called **leaves**.

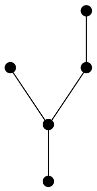
Rooted Trees Drawing Convention

Throughout this presentation the following convention is used:

- vertices are drawn without labels,
- the root is drawn at the bottom,
- child vertices are drawn above their respective parent vertex.



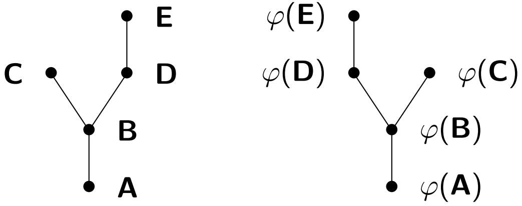
is drawn as



Abstract Rooted Trees

Two rooted trees are regarded as being **equivalent** if there exists an order isomorphism between their vertex sets, that is, a bijection that preserves edge relations.

Example:



Abstract Rooted Trees (cont.)

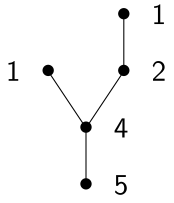
Thus each rooted tree belongs to a unique equivalence class with respect to this notion of equivalence.

- For a given rooted tree, its **abstract rooted tree** is its equivalence class.
- The letter $\mathbf{t}$ is used to denote an arbitrary rooted tree and $|\mathbf{t}|$ denotes the corresponding abstract rooted tree.
- The set of all rooted trees is denoted $\mathbf{T}$.

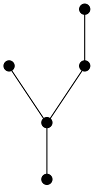
Important Functions on Trees

For a given rooted tree **t**, the following values are of interest:

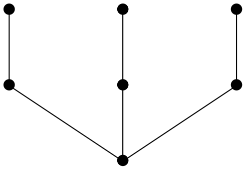
- $r(t)$: the **order** or number of vertices in the tree,
- $\sigma(t)$: the **symmetry** or the number of automorphisms for the tree, (that is, the number of bijections from the tree's vertex set to itself that preserve the ordering scheme,)
- $\gamma(t)$: the **density** of the tree, (this is defined to be the value obtained by taking the order of each subtree that would be formed if each vertex of the tree were the root (not counting the vertices not at or above the vertex) and then multiplying all of these values together.)



Important Functions on Trees (examples)



$r(t) = 5, \sigma(t) = 1, \gamma(t) = 40$



$r(t) = 7, \sigma(t) = 6, \gamma(t) = 56$

More Functions on Trees

$$\alpha(t) := \frac{r(t)!}{\sigma(t)\gamma(t)} (= \frac{\beta(t)}{\gamma(t)})$$

$$\beta(t) := \frac{r(t)!}{\sigma(t)}$$

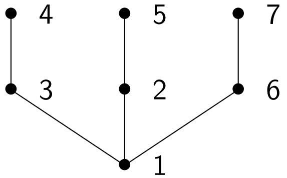
- $\alpha(t)$: the number of ways to label **t**'s vertices using elements from the set $\{1, 2, \ldots, r(t)\}$ such that

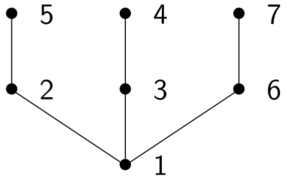
  1. each vertex receives exactly one label (repetition is not allowed,)
  2. labelings that indicate an automorphism are counted only once,
  3. if (a, b) is a directed edge in t's edge set, then the label for a is less than the label for b.

- $\beta(t)$: same as $\alpha(t)$ but without the third condition.

What Condition 2. Means

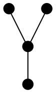
Condition 2. states that the following are counted as the same labeling:



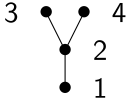


Examples for $\alpha(t)$, $\beta(t)$

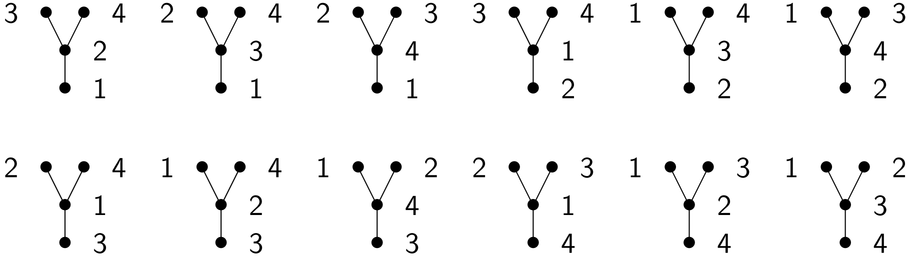
Tree:



α **Labelings:**



β **Labelings:**



Notation for Rooted Trees

A rooted tree with a single vertex, that is, a tree with a root vertex and nothing else, is denoted by τ.

Other rooted trees are denoted using a recursive notation. In general,

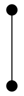
$$[t_1, t_2, \ldots, t_m]$$

denotes a rooted tree where the root has m child nodes which form the respective subtrees $t_1, t_2, \ldots, t_m$.

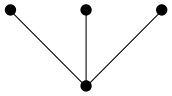
Since a subtree can show up more than once, it is convenient to use powers in the notation to indicate repetition. An arbitrary rooted tree can also be represented as

$$[t_1^{m_1}, t_2^{m_2}, \ldots, t_n^{m_n}]$$

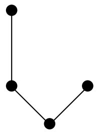
Notation for Rooted Trees (examples)



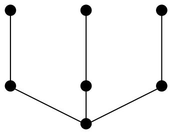
$[\tau]$



$\left[\tau^3\right]$



$[[\tau]\,\tau]$



$\left[[\tau]^3\right]$

Some Basic Results

Theorem

Let $\mathbf{t} = [t_1^{m_1}, t_2^{m_2}, \ldots, t_n^{m_n}]$ *be a rooted tree, then*

1. $r(t) = 1 + \sum_{i=1}^{n} m_i r(t_i)$
2. $\sigma(t) = \prod_{i=1}^{n} m_i! \sigma(t_i)^{m_i}$
3. $\gamma(t) = r(t) \prod_{i=1}^{n} \gamma(t_i)^{m_i}$

Also recall that

4. $\alpha(t) = \frac{r(t)!}{\sigma(t)\gamma(t)}$
5. $\beta(t) = \frac{r(t)!}{\sigma(t)}$

Numerical Analysis with Rooted Trees

Now that we know a bit about rooted trees and their properties, we can use them to solve our problem in numerical analysis, that is,

"How can we derive the system of equations for Runge-Kutta methods of a desired order without having to take so many Taylor expansions?"

We will see that both the Taylor expansion

$$y(x_n + h) = y(x_n) + hy'(x_n) + \frac{h^2}{2!}y''(x_n) + \dots$$

and the Runge-Kutta approximation

$$y_{n+1} = y_n + b_1 hf(Y_1) + b_2 hf(Y_2) + \cdots + b_s hf(Y_s)$$

can be expressed in terms of rooted trees.

Outline

The Elementary Differential

Assume that we have an autonomous ordinary differential equation, that is

$$y'(x) = f(y(x)) \text{ for all } x \in [a, b], \text{ and } y(x_0) = y_0.$$

(Note that $y(x)$ can be an ordinary function or a vector.)

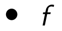
Definition

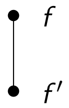
Given a tree **t** and a function $f : R^N \to R^N$ analytic in some neighbourhood of y, the **Elementary Differential** $F(t)(y)$ is defined recursively by

$$F(\tau)(y) = f(y)$$

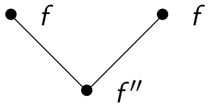
$$F([t_1, t_2, \ldots, t_m])(y) = f^{(m)}(y)(F(t_1)(y), F(t_2)(y), \ldots, F(t_m)(y))$$

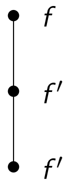
The Elementary Differential (examples)





Note that $\frac{d}{dx} f(y(x)) = f'(y(x)) f(y(x))$.





Similarly, $\frac{d^2}{dx^2} f(y(x)) = F([\tau^2])(y(x)) + F([[\tau]])(y(x))$.

Taking the Derivative of an Elementary Differential

Lemma

Let $\mathbf{t}$ *be a rooted tree, then*

$$\frac{d}{dx}F(|t|)(y)$$

is the sum of $F(u)(y)$ *over all rooted trees* $\mathbf{u}$ *such that if a leaf vertex is removed from* $\mathbf{u}$, *it becomes* $\mathbf{t}$.

Taking the Derivative of an Elementary Differential

Lemma

Let $\mathbf{t}$ *be a rooted tree, then*

$$\frac{d}{dx}F(|t|)(y)$$

is the sum of $F(u)(y)$ *over all rooted trees* $\mathbf{u}$ *such that if a leaf vertex is removed from* $\mathbf{u}$*, it becomes* $\mathbf{t}$*.*

Sketch Proof:

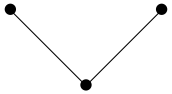
This can be proved by complete induction on the order of $\mathbf{t}$ (on $n = r(t)$). A key step is in how if we let $\mathbf{t} = [t_1, t_2, \ldots t_m]$, then

$$\frac{d}{dx}F(|t|)(y(x)) = \frac{d}{dx}(f^{(m)}(y)(F(t_1)(y), F(t_2)(y), \ldots, F(t_m)(y)))$$

The extended product rule can then be used. The $f^{(m+1)}(y)$ term corresponds to adding a leaf to the root. The rest of the terms follow from the inductive hypothesis.

Taking the Derivative of an Elementary Differential (cont.)

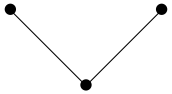
An important thing to note about the lemma is that it is actually the location of the inserted leaf vertex that is summed over. It does not matter if two different locations of the inserted leaf result in trees that are equivalent.



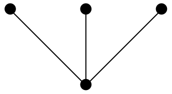
$$F([\tau^2])(y(x)) = f''(y(x))f(y(x))^2$$

Taking the Derivative of an Elementary Differential (cont.)

An important thing to note about the lemma is that it is actually the location of the inserted leaf vertex that is summed over. It does not matter if two different locations of the inserted leaf result in trees that are equivalent.

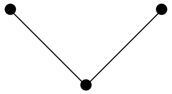


$$F([\tau^2])(y(x)) = f''(y(x))f(y(x))^2$$

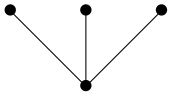


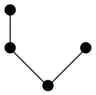
Taking the Derivative of an Elementary Differential (cont.)

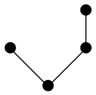
An important thing to note about the lemma is that it is actually the location of the inserted leaf vertex that is summed over. It does not matter if two different locations of the inserted leaf result in trees that are equivalent.



$$F([\tau^2])(y(x)) = f''(y(x))f(y(x))^2$$

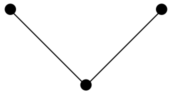




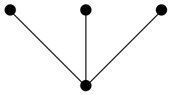


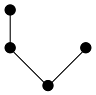
Taking the Derivative of an Elementary Differential (cont.)

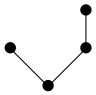
An important thing to note about the lemma is that it is actually the location of the inserted leaf vertex that is summed over. It does not matter if two different locations of the inserted leaf result in trees that are equivalent.



$$F([\tau^2])(y(x)) = f''(y(x))f(y(x))^2$$







$$\frac{d}{dx}F([\tau^2])(y(x)) = F([\tau^3])(y(x)) + 2F([[\tau]\,\tau])(y(x))$$

Obtaining Derivatives for the Taylor expansion

Theorem

Let $y(x)$ be a function such that $y'(x) = f(y(x))$, then assuming that it exists,

$$y^{(k)}(x) = \sum_{r(|t|)=k} \alpha(|t|)F(|t|)(y(x))$$

That is, the k^{th} derivative of y can be found using a representative of each equivalence class from the set all rooted trees with k vertices.

Obtaining Derivatives for the Taylor expansion

Theorem

Let $y(x)$ be a function such that $y'(x) = f(y(x))$, then assuming that it exists,

$$y^{(k)}(x) = \sum_{r(|t|)=k} \alpha(|t|)F(|t|)(y(x))$$

That is, the k^{th} derivative of y can be found using a representative of each equivalence class from the set all rooted trees with k vertices.

Sketch Proof:
By applying the preceding lemma repeatedly, it is clear that we will be summing all rooted trees of order k, however some rooted trees will be appearing more than once. This is where $\alpha(t)$ comes into play.

An alternative way to interpret $\alpha(t)$ is as the number of orders in which you "can add vertices to the top" so that $|\mathbf{t}|$ is constructed.

Obtaining the Taylor Expansion

Theorem

The Taylor expansion of $y(x)$ about $x = x_n$ (assuming it exists) is

$$y(x_n + h) = y(x_n) + \sum_{|t| \in T} \frac{h^{r(|t|)}}{\sigma(|t|)\gamma(|t|)} F(|t|)(y(x_n))$$

Obtaining the Taylor Expansion

Theorem

The Taylor expansion of $y(x)$ about $x = x_n$ (assuming it exists) is

$$y(x_n + h) = y(x_n) + \sum_{|t| \in T} \frac{h^{r(|t|)}}{\sigma(|t|)\gamma(|t|)} F(|t|)(y(x_n))$$

Proof:

$$y(x_n+h) = y(x_n)+\sum_{k=1}^{\infty} \frac{h^k}{k!} y^{(k)}(x_n) = y(x_n)+\sum_{k=1}^{\infty} \frac{h^k}{k!} \sum_{r(|t|)=k} \alpha(|t|)F(|t|)(y(x_n))$$

$$= y(x_n) + \sum_{|t| \in T} \frac{h^{r(|t|)}\alpha(|t|)}{r(|t|)!} F(|t|)(y(x_n))$$

Recalling that $\alpha(t) = \frac{r(t)!}{\sigma(t)\gamma(t)}$, the result follows.

Obtaining the Taylor Expansion (cont.)

Corollary

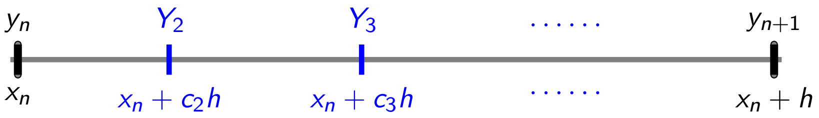
For any natural number P,

$$y(x_n + h) = y(x_n) + \sum_{r(|t|)\leq P} \frac{1}{\gamma(|t|)} \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y(x_n)) + O(h^{P+1})$$

This gives us the truncated Taylor expansion of the exact solution to the ODE in terms of rooted trees.

Outline

A Brief Recap of Runge-Kutta



For an explicit Runge-Kutta method with s stages we need to determine

$$X_1 = x_n, X_2 = x_n + \mathbf{c_2} h, X_3 = x_n + \mathbf{c_3} h, \ldots, X_s = x_n + \mathbf{c_s} h$$

We also need to know how each stage depends on the previously computed stage derivatives.

$$Y_1 = y_n, Y_2 = y_n + \mathbf{a_{21}} hf(Y_1), \ldots, Y_s = y_n + \sum_{j=1}^{s-1} \mathbf{a_{sj}} hf(Y_j)$$

Lastly we need to know the weights

$$y_{n+1} = y_n + \mathbf{b_1} hf(Y_1) + \mathbf{b_2} hf(Y_2) + \ldots \mathbf{b_s} hf(Y_s)$$

The Weight Functions

Definition

Let

$$\begin{array}{c|c} c & A \\ \hline & b \end{array}$$

denote the tableau for an explicit Runge-Kutta method with s stages.

The Weight Functions

Definition

Let

$$\begin{array}{c|c} c & A \\ \hline & b \end{array}$$

denote the tableau for an explicit Runge-Kutta method with s stages. Then the **elementary weights** $\Phi(t)$, **internal weights** $\Phi_i(t)$, and **derivative weights** $\Phi_i^D(t)$ for $t \in T$ and $i = 1, 2, \ldots, s$ are defined by

$$\Phi(t) = \textstyle\sum_{i=1}^{s} b_i \Phi_i^D(t), \qquad \Phi_i(t) = \textstyle\sum_{j=1}^{i-1} a_{ij} \Phi_j^D(t),$$

$$\Phi_i^D(t) = \begin{cases} 1 & : t = \tau \\ \prod_{j=1}^{m} \Phi_i(t_j) & : t = [t_1, t_2, \ldots, t_m] \end{cases}$$

Weights and Internal Weights look the same

$$\Phi_i(t) = \sum_{j=1}^{i-1} a_{ij}\Phi_j^D(t) \text{ comes from } Y_i = y_n + \sum_{j=1}^{i-1} a_{ij} h f(Y_j),$$

and

$$\Phi(t) = \sum_{i=1}^{s} b_i\Phi_i^D(t) \text{ comes from } y_{n+1} = y_n + \sum_{i=1}^{s} b_i h f(Y_i).$$

What's interesting to note is that the approximation for $y(x_n + h)$ is computed in a way that is similar to the way that stages are computed. The calculation for y_{n+1} can be thought of as a final stage.

Where the Derivative Weights come from

The motivation for the derivative weights comes from the following result:

Lemma

The Taylor expansion for

$$hf\left(y_n + \sum_{|t|\in T} \Phi_i(|t|)\frac{h^{r(|t|)}}{\sigma(|t|)}F(|t|)(y_n)\right)$$

is

$$\sum_{|t|\in T} \Phi_i^D(|t|)\frac{h^{r(|t|)}}{\sigma(|t|)}F(|t|)(y_n)$$

The proof of this result is not appropriate for a short presentation as there will be many things going on at once. If you are interested, we can talk about it afterwards.

The Taylor Expansion for the Runge-Kutta Approximation

Theorem

The truncated Taylor expansions for the stages, stage derivatives, and approximation for an explicit s-stage Runge-Kutta method of order P are

$$Y_i = y_n + \sum_{r(|t|)\le P} \Phi_i(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n) + O(h^{P+1}),$$

$$hf(Y_i) = \sum_{r(|t|)\le P} \Phi_i^D(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n) + O(h^{P+1}),$$

(where $i = 1, 2, \dots, s$)

$$y_{n+1} = y_n + \sum_{r(|t|)\le P} \Phi(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n) + O(h^{P+1})$$

The Taylor Expansion for the Runge-Kutta Approximation (proof)

Proof:
The proof is by complete induction on the stage number.

$$\Phi_1(t) = 0 \text{ for all } t \in T \implies Y_1 = y_n$$

$$\Phi_1^D(t) = \begin{cases} 1 & : t = \tau \\ 0 & : otherwise \end{cases} \implies hf(Y_1) = \frac{h^{r(|\tau|)}}{\sigma(|\tau|)} f(y_n)$$

The base case clearly holds.

The Taylor Expansion for the Runge-Kutta Approximation (proof cont.)

Suppose that for all $1 \leq i \leq k$, the result holds, that is

$$Y_i = y_n + \sum_{r(|t|)\leq P} \Phi_i(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n) + O(h^{P+1}),$$

then applying the most recent lemma gives us (for all $1 \leq i \leq k$)

$$hf(Y_i) = \sum_{r(|t|)\leq P} \Phi_i^D(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n) + O(h^{P+1}).$$

The Taylor Expansion for the Runge-Kutta Approximation (proof cont.)

The next stage is computed as follows

$$Y_{k+1} = y_n + \sum_{j=1}^{k} a_{(k+1)j} h f(Y_j)$$

$$= y_n + \sum_{r(|t|)\leq P} \sum_{j=1}^{k} a_{(k+1)j} \Phi_j^D(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n) + O(h^{P+1})$$

$$= y_n + \sum_{r(|t|)\leq P} \Phi_{k+1}(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n) + O(h^{P+1})$$

Thus the inductive hypothesis holds. The expansion for y_{n+1} is just a special case where the a_{ij} terms are replaced by b_i. By induction, we have our result.

Outline

Equating the Expansions

Assume that we have an autonomous ordinary differential equation,

$$y'(x) = f(y(x)) \text{ for all } x \in [a, b], y(x_0) = y_0$$

An s-stage order P Runge-Kutta method will require the terms of the truncated Taylor expansion for the exact solution

$$y(x_n + h) \approx y(x_n) + \sum_{r(|t|) \leq P} \frac{1}{\gamma(|t|)} \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y(x_n))$$

to equate with the terms of the truncated Taylor expansion for the approximation

$$y_{n+1} = y_n + \sum_{r(|t|) \leq P} \Phi(|t|) \frac{h^{r(|t|)}}{\sigma(|t|)} F(|t|)(y_n).$$

Equating the Expansions (cont.)

This is accomplished if

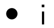
$$\Phi(|t|) = \frac{1}{\gamma(|t|)} \text{ for all } |t| \text{ such that } r(|t|) \leq P$$

In other words, the equations that the a_{ij}, b_i, and c_j terms must satisfy can be obtained without doing any Taylor expansions!

Instead, we can take a representative of each equivalence class from the set of rooted trees with order $\leq P$ and equate $\Phi(|t|) = \frac{1}{\gamma(|t|)}$ for each one.

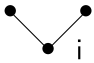
An Easier Way to Compute $\Phi(|t|)$

- At the root write b_i in the sum. Label the root with the letter i.



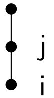
b_i

- At leaf vertices write c_* in the sum where $*$ is the label of the leaf's parent.



$b_i c_i c_i$

- At all other vertices write a_{*-} in the sum where $*$ is the label of the vertex's parent and $-$ is a letter not yet used as a label.
  Label the vertex with $-$.

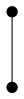


$b_i a_{ij} c_j$

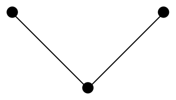
Deriving the Third Order methods with 3 stages

The rooted trees with order ≤ 3 and corresponding equations are

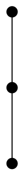
$\sum_{i=1}^{s} b_i = 1$



$\sum_{i=1}^{s} b_i c_i = 1/2$



$\sum_{i=1}^{s} b_i c_i^2 = 1/3$



$\sum_{i=1}^{s} \sum_{j=1}^{i-1} b_i a_{ij} c_j = 1/6$

Deriving the Third Order methods with 3 stages (cont.)

We thus have the system of equations

$$b_1 + b_2 + b_3 = 1$$

$$b_2 c_2 + b_3 c_3 = 1/2$$
$$b_2 c_2^2 + b_3 c_3^2 = 1/3$$
$$b_3 a_{32} c_2 = 1/6$$

We can now let c_2 and c_3 be free parameters in order to make the first few equations into a linear system.

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & c_2 & c_3 \\ 0 & c_2^2 & c_3^2 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1/2 \\ 1/6 \end{bmatrix}$$

and then deal with the fourth equation once b_1, b_2, b_3 are known.

Deriving the Third Order methods with 3 stages (cont.)

The determinant of the coefficient matrix in the linear system is

$$c_2 c_3 (c_3 - c_2)$$

And the fourth equation was

$$b_3 a_{32} c_2 = 1/6$$

There are three cases where a solution exists:

1. $c_2 c_3 (c_3 - c_2) \neq 0$
2. $c_2 = 2/3, c_3 = 0, b_3 \neq 0$
3. $c_2 = 2/3, c_3 = 2/3, b_3 \neq 0$

Outline

The Motivation for Imposing Conditions

$$\Phi(|t|) = \frac{1}{\gamma(|t|)} \text{ for all } |t| \text{ such that } r(|t|) \leq P$$

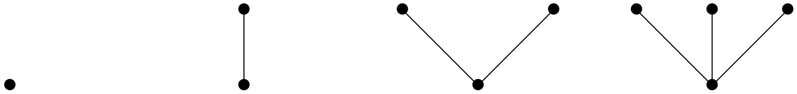
Thus far we have figured out a way to obtain the system of equations without doing any Taylor expansions.

However, as the order of the methods we wish to derive increases, so does the number of rooted trees. In fact, the number of rooted trees we must deal with seems to grow exponentially as the sought after order increases.

The idea now is to impose simplifying conditions that the coefficients need to satisfy so that we have to worry about less rooted trees.

The $B(\eta)$ Conditions

The following rooted trees all have a similar form, that is, they all consist of only a root with leaves.



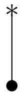
For a Runge-Kutta method to have order P, we would normally have to worry about P of these trees. We can instead impose the condition

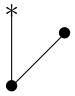
$$\sum_{i=1}^{s} b_i c_i^{k-1} = 1/k \text{ for all } k = 1, 2, \ldots, \eta$$

and then set η equal to P.

The $D(1)$ Condition

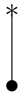
Suppose that we have the equations holding for two rooted trees of the following form

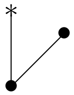




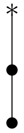
The $D(1)$ Condition

Suppose that we have the equations holding for two rooted trees of the following form

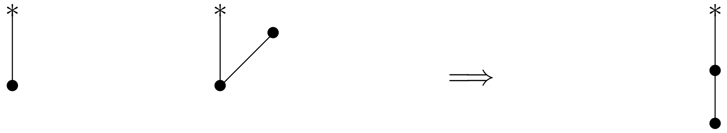




If the $D(1)$ condition holds, then the equation for the following tree is automatically satisfied



What the $D(1)$ Condition is



Let's call these trees (from left to right) $\mathbf{t_1}, \mathbf{t_2}, \mathbf{t_3}$. Then in general,

$$\frac{1}{\gamma(t_1)} - \frac{1}{\gamma(t_2)} = \frac{1}{\gamma(t_3)}$$

Thus we want to force

$$\sum_j b_j * - \sum_j b_j c_j * = \sum_j \sum_i b_i a_{ij} *$$

What the $D(1)$ Condition is (cont.)

The $D(1)$ condition states that

$$b_j(1-c_j) = \sum_{i=1}^{s} b_i a_{ij} \text{ for any } j = 1, 2, \ldots, s$$

What the $D(1)$ Condition is (cont.)

The $D(1)$ condition states that

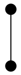
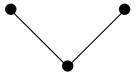
$$b_j(1 - c_j) = \sum_{i=1}^{s} b_i a_{ij} \text{ for any } j = 1, 2, \ldots, s$$

In general, the $D(k)$ condition states that (for any $j = 1, 2, \ldots, s$)

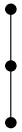
$$\frac{1}{q} b_j(1 - c_j^q) = \sum_{i=1}^{s} b_i c_i^{q-1} a_{ij} \text{ for all } q = 1, 2, \ldots, k$$

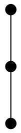
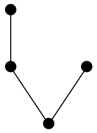
but for explicit Runge-Kutta methods, only the $D(1)$ condition can be used as the $D(2)$ condition leads to an inconsistent system of equations.

D(1) Condition Examples

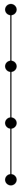



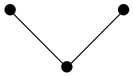
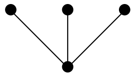
$\Longrightarrow$



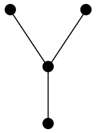



$\Longrightarrow$



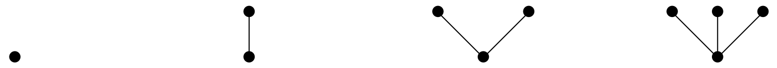



$\Longrightarrow$

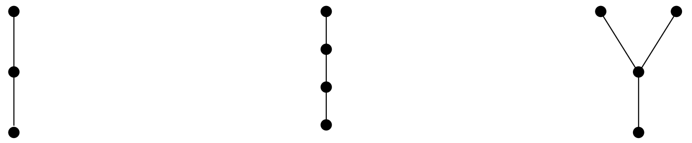


Deriving the Fourth Order methods with 4 stages

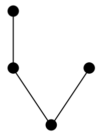
The $B(4)$ condition takes care of



The $D(1)$ condition takes care of

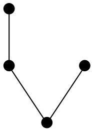


What's left?



Deriving the Fourth Order methods with 4 stages (cont.)

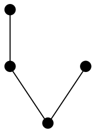
To deal with



$$\sum_{i=1}^{4}\sum_{j=1}^{i-1} b_i c_i a_{ij} c_j = 1/8$$

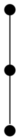
Deriving the Fourth Order methods with 4 stages (cont.)

To deal with



$$\sum_{i=1}^{4}\sum_{j=1}^{i-1} b_i c_i a_{ij} c_j = 1/8$$

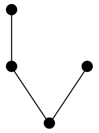
we can subtract it from



$$\sum_{i=1}^{4}\sum_{j=1}^{i-1} b_i a_{ij} c_j = 1/6$$

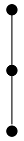
Deriving the Fourth Order methods with 4 stages (cont.)

To deal with



$$\sum_{i=1}^{4}\sum_{j=1}^{i-1} b_i c_i a_{ij} c_j = 1/8$$

we can subtract it from



$$\sum_{i=1}^{4}\sum_{j=1}^{i-1} b_i a_{ij} c_j = 1/6$$

To get the more convenient equation

$$\sum_{i=1}^{4}\sum_{j=1}^{i-1} b_i(1-c_i) a_{ij} c_j = b_3(1-c_3)a_{32}c_2 = 1/24$$

Deriving the Fourth Order methods with 4 stages (cont.)

We thus have the system of equations

$$\sum_{i=1}^{4} b_i c_i^{k-1} = 1/k \text{ for all } k = 1, 2, \ldots, 4$$

$$b_j(1 - c_j) = \sum_{i=1}^{4} b_i a_{ij} \text{ for any } j = 1, 2, \ldots, 4$$

$$b_3(1 - c_3) a_{32} c_2 = 1/24$$

Deriving the Fourth Order methods with 4 stages (cont.)

We thus have the system of equations

$$\sum_{i=1}^{4} b_i c_i^{k-1} = 1/k \text{ for all } k = 1, 2, \ldots, 4$$

$$b_j(1 - c_j) = \sum_{i=1}^{4} b_i a_{ij} \text{ for any } j = 1, 2, \ldots, 4$$

$$b_3(1 - c_3)a_{32}c_2 = 1/24$$

A nice consequence of the $D(1)$ condition is that it immediately implies that $c_4 = 1$.

Deriving the Fourth Order methods with 4 stages (cont.)

A fourth order method with 4 stages can thus be constructed as follows:

Deriving the Fourth Order methods with 4 stages (cont.)

A fourth order method with 4 stages can thus be constructed as follows:

- Choose values for c_2 and c_3, noting that $c_1 = 0$ and $c_4 = 1$.

Deriving the Fourth Order methods with 4 stages (cont.)

A fourth order method with 4 stages can thus be constructed as follows:

- Choose values for c_2 and c_3, noting that $c_1 = 0$ and $c_4 = 1$.
- Obtain b_1, b_2, b_3, b_4 by solving the linear system

$$\sum_{i=1}^{4} b_i c_i^{k-1} = 1/k \text{ for all } k = 1, 2, \ldots, 4$$

Deriving the Fourth Order methods with 4 stages (cont.)

A fourth order method with 4 stages can thus be constructed as follows:

- Choose values for c_2 and c_3, noting that $c_1 = 0$ and $c_4 = 1$.
- Obtain b_1, b_2, b_3, b_4 by solving the linear system

$$\sum_{i=1}^{4} b_i c_i^{k-1} = 1/k \text{ for all } k = 1, 2, \ldots, 4$$

- Set $a_{32} = \frac{1}{24 b_3 (1 - c_3) c_2}$, $a_{31} = c_3 - a_{32}$, $a_{21} = c_2$.

Deriving the Fourth Order methods with 4 stages (cont.)

A fourth order method with 4 stages can thus be constructed as follows:

- Choose values for c_2 and c_3, noting that $c_1 = 0$ and $c_4 = 1$.
- Obtain b_1, b_2, b_3, b_4 by solving the linear system

$$\sum_{i=1}^{4} b_i c_i^{k-1} = 1/k \text{ for all } k = 1, 2, \ldots, 4$$

- Set $a_{32} = \frac{1}{24 b_3 (1-c_3) c_2}$, $a_{31} = c_3 - a_{32}$, $a_{21} = c_2$.
- Obtain a_{41}, a_{42}, a_{43} by using the $D(1)$ equations

$$a_{4j} = \frac{b_j(1-c_j) - \sum_{i=1}^{3} b_i a_{ij}}{b_4} \text{ for } j = 1, 2, 3$$

Deriving the Fourth Order methods with 4 stages (cont.)

A fourth order method with 4 stages can thus be constructed as follows:

- Choose values for c_2 and c_3, noting that $c_1 = 0$ and $c_4 = 1$.
- Obtain b_1, b_2, b_3, b_4 by solving the linear system

$$\sum_{i=1}^{4} b_i c_i^{k-1} = 1/k \text{ for all } k = 1, 2, \ldots, 4$$

- Set $a_{32} = \frac{1}{24 b_3 (1-c_3) c_2}$, $a_{31} = c_3 - a_{32}$, $a_{21} = c_2$.
- Obtain a_{41}, a_{42}, a_{43} by using the $D(1)$ equations

$$a_{4j} = \frac{b_j(1 - c_j) - \sum_{i=1}^{3} b_i a_{ij}}{b_4} \text{ for } j = 1, 2, 3$$

There are some subtleties but by following this approach, we can get our fourth order Runge-Kutta!

Concluding Remarks

Something that's interesting to note is that the approach of using rooted trees to derive Runge-Kutta methods discussed in this presentation more or less applies to implicit Runge-Kutta methods as well.

A reason why implicit methods may be of interest is because explicit Runge-Kutta methods can never be A-stable.

If time permits, the final report may also include use of Runge-Kutta to aid in solving a problem related to Financial Math. (That being computing the portfolios on the Efficient Frontier.)

Thank you for viewing my presentation. ☺

How to Prove the Scary Lemma

Remember this?

Lemma

The Taylor expansion for

$$hf\left(y_n + \sum_{|t|\in T} \Phi_i(|t|)\frac{h^{r(|t|)}}{\sigma(|t|)}F(|t|)(y_n)\right)$$

is

$$\sum_{|t|\in T} \Phi_i^D(|t|)\frac{h^{r(|t|)}}{\sigma(|t|)}F(|t|)(y_n)$$

Read on for instructions on how to prove it.

How to Prove the Scary Lemma (cont.)

We will need to use the following general result

Theorem

$$f(y+\sum_{i=1}^{m}\delta_i)=\sum_{I\in I_m}\frac{1}{\sigma(I)}f^{(\#I)}(y)\delta_I$$

Where

- I_m is the collection of finite sequences of the form $(i_1, i_2, \dots, i_n)$ where each term is of the set {1,2, . . . , m}. (The empty sequence () is included.)
- $\delta_I = (\delta_{i_1}, \delta_{i_2}, \dots, \delta_{i_n})$ for $I = (i_1, i_2, \dots, i_n)$.
- $\sigma(I)$ is the number of ways of permuting the elements of the sequence without changing it.

$$\sigma(I) = k_1!k_2!\dots k_m!$$

where k_j is the number of times j appears in the sequence.

How to Prove the Scary Lemma (cont.)

The idea is that with the rooted trees, there is a finite number of trees of a given order so T is a countably infinite set and we can write

$$T = \{t_1, t_2, t_3, \dots\}$$

The sequence $I = (i_1, i_2, \dots, i_n)$ will correspond with $(t_{i_1}, t_{i_2}, \dots t_{i_n})$. This means

$$\frac{1}{\sigma(I)} f^{(\#I)}(y)\delta_I = f^{(n)}(y)\frac{\Phi_i(t_{i_1})F(t_{i_1})(y)\Phi_i(t_{i_2})F(t_{i_2})(y)\dots\Phi_i(t_{i_n})F(t_{i_n})(y)}{\sigma(I)\sigma(t_{i_1})\sigma(t_{i_2})\dots\sigma(t_{i_n})}$$

$$= \frac{\Phi_i^D([t_{i_1}, t_{i_2}, \dots t_{i_n}])F([t_{i_1}, t_{i_2}, \dots t_{i_n}])(y)}{\sigma([t_{i_1}, t_{i_2}, \dots t_{i_n}])}$$

The above is a sketch and does not show everything that is going on. It does not show the $r(t_{i_j})$ multiplying together to form $r([t_{i_1}, t_{i_2}, \dots t_{i_n}]) - 1$ and is for the general case when $t \neq \tau$ (τ corresponds to $I = ()$ and is trivial.)