Spreadsheet drawings of plant branching from modified Lindenmayer grammars

John Banks

The University of Melbourne

October 9, 2015

The Context

MAT1MAB Mathematical Applications in Biology (La Trobe University):

- No calculus assumed or taught.
- Organised around application focussed themes (Learning Modules).
- Uses spreadsheets to run simulations/models (this made the biologists happy).
- No lectures. Two 1 hour practice classes per week.

The Context

MAT1MAB Mathematical Applications in Biology (La Trobe University):

- No calculus assumed or taught.
- Organised around application focussed themes (Learning Modules).
- Uses spreadsheets to run simulations/models (this made the biologists happy).
- No lectures. Two 1 hour practice classes per week.
- Objectives(?):
  - Give the flavour of some accessible models.
  - Teach spreadsheet skills.
  - Surreptitiously improve quantitative/algebraic skills.

The Content

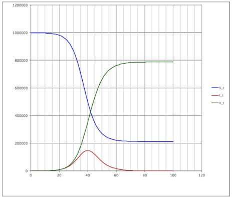
- Six two week Learning Modules:
  1. Growth and Scaling:
     Size matters and matters of size.
  2. Epidemics: Enough to make you sick!
  3. **Cellular Automata:**
     What do epidemics and bushfires have in common?
  4. Population Models: When time runs smoothly.
  5. Growth Creates Form: The shape of things organic.
  6. Climate Models: Taking Earth's temperature.

The Content

- Six two week Learning Modules:
  1. Growth and Scaling:
     Size matters and matters of size.
     Real agenda: Revise basic functions.
     
  2. Epidemics: Enough to make you sick!
  3. **Cellular Automata:**
     What do epidemics and bushfires have in common?
  4. Population Models: When time runs smoothly.
  5. Growth Creates Form: The shape of things organic.
  6. Climate Models: Taking Earth's temperature.

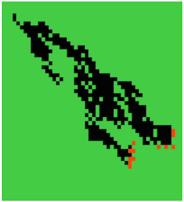
The Content

- Six two week Learning Modules:
  1. Growth and Scaling:
     Size matters and matters of size.
  2. Epidemics: Enough to make you sick!
     Real agenda: Intro to difference equations



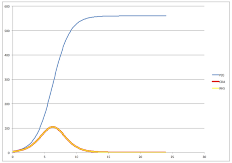
  3. **Cellular Automata:**
     What do epidemics and bushfires have in common?
  4. Population Models: When time runs smoothly.
  5. Growth Creates Form: The shape of things organic.
  6. Climate Models: Taking Earth's temperature.

The Content

- Six two week Learning Modules:
  1. Growth and Scaling:
     Size matters and matters of size.
  2. Epidemics: Enough to make you sick!
  3. **Cellular Automata:**
     What do epidemics and bushfires have in common?
     Real agenda: Cellular Automata!?
     
  4. Population Models: When time runs smoothly.
  5. Growth Creates Form: The shape of things organic.
  6. Climate Models: Taking Earth's temperature.

The Content

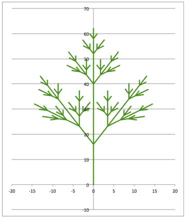
- Six two week Learning Modules:
  1. Growth and Scaling:
     Size matters and matters of size.
  2. Epidemics: Enough to make you sick!
  3. **Cellular Automata:**
     What do epidemics and bushfires have in common?
  4. Population Models: When time runs smoothly.
     Real agenda: Differential equations (by stealth).



  5. Growth Creates Form: The shape of things organic.
  6. Climate Models: Taking Earth's temperature.

The Content

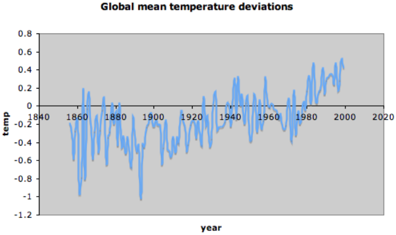
- Six two week Learning Modules:
  1. Growth and Scaling:
     Size matters and matters of size.
  2. Epidemics: Enough to make you sick!
  3. **Cellular Automata:**
     What do epidemics and bushfires have in common?
  4. Population Models: When time runs smoothly.
  5. Growth Creates Form: The shape of things organic.
     Real agenda: Motivate trig functions and algorithmic thinking (and have fun!)

  6. Climate Models: Taking Earth's temperature.

The Content

- Six two week Learning Modules:
  1. Growth and Scaling:
     Size matters and matters of size.
  2. Epidemics: Enough to make you sick!
  3. **Cellular Automata:**
     What do epidemics and bushfires have in common?
  4. Population Models: When time runs smoothly.
  5. Growth Creates Form: The shape of things organic.
  6. Climate Models: Taking Earth's temperature.
     Real agenda: Intro to stochastic processes.



The Grammars

- Lindenmayer Grammars can be used to model various geometric and biological phenomena.
- Motivating application of branching in annual plants is a nice example of a mathematical model in biology.
- A (context free) Lindenmayer Grammar specifies the branching rules for a plant.

The Grammars

- Lindenmayer Grammars can be used to model various geometric and biological phenomena.
- Motivating application of branching in annual plants is a nice example of a mathematical model in biology.
- A (context free) Lindenmayer Grammar specifies the branching rules for a plant.
- A Lindenmayer Grammar (L-System) has an alphabet, a start string w_0 and some rewrite rules.

The Grammars

- Lindenmayer Grammars can be used to model various geometric and biological phenomena.
- Motivating application of branching in annual plants is a nice example of a mathematical model in biology.
- A (context free) Lindenmayer Grammar specifies the branching rules for a plant.
- A Lindenmayer Grammar (L-System) has an alphabet, a start string w_0 and some rewrite rules.
- Derivations from this grammar generate an abstract instruction strings $w_1, w_2, w_3, \ldots$.

The Grammars

- Lindenmayer Grammars can be used to model various geometric and biological phenomena.
- Motivating application of branching in annual plants is a nice example of a mathematical model in biology.
- A (context free) Lindenmayer Grammar specifies the branching rules for a plant.
- A Lindenmayer Grammar (L-System) has an alphabet, a start string w_0 and some rewrite rules.
- Derivations from this grammar generate an abstract instruction strings $w_1, w_2, w_3, \ldots$.
- Each w_i describes the branching structure at a particular stage of growth.

The Example

- Alphabet: $X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Rules:

$$r_0 : \; X \to F[+X][-X]FX \quad \big(\text{Replace all } X\text{'s by } F[+X][-X]FX\big)$$
$$r_1 : \; F \to FF \quad \big(\text{Replace all } F\text{'s by } FF\big)$$

- No rules for $[,], +, -$, so leave them alone.
- No need for $] \to]$ rules, etc.

Example Derivation

$w_0 = X$

The Example

- Alphabet: $X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Rules:

$$r_0 : \ X \to F[+X][-X]FX \quad \big(\text{Replace all } X\text{'s by } F[+X][-X]FX\big)$$
$$r_1 : \ F \to FF \quad \big(\text{Replace all } F\text{'s by } FF\big)$$

- No rules for $[,], +, -$, so leave them alone.
- No need for $] \to]$ rules, etc.

Example Derivation

$$w_0 = X$$
$$w_1 = F[+X][-X]FX$$

The Example

- Alphabet: $X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Rules:

$$r_0 : \; X \to F[+X][-X]FX \quad \big(\text{Replace all } X\text{'s by } F[+X][-X]FX\big)$$

$$r_1 : \; F \to FF \quad \big(\text{Replace all } F\text{'s by } FF\big)$$

- No rules for $[,], +, -$, so leave them alone.
- No need for $] \to]$ rules, etc.

Example Derivation

$$w_0 = X$$

$$w_1 = F[+X][-X]FX$$

$$w_2 = FF[+F[+X][-X]FX][-F[+X][-X]FX]FFF[+X][-X]FX$$

The Example

- Alphabet: $X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Rules:

$$r_0 : \ X \to F[+X][-X]FX \quad \big(\text{Replace all } X\text{'s by } F[+X][-X]FX\big)$$
$$r_1 : \ F \to FF \quad \big(\text{Replace all } F\text{'s by } FF\big)$$

- No rules for $[,], +, -$, so leave them alone.
- No need for $] \to]$ rules, etc.

Example Derivation

$w_0 = X$

$w_1 = F[+X][-X]FX$

$w_2 = FF[+F[+X][-X]FX][-F[+X][-X]FX]FFF[+X][-X]FX$

$w_3 =$ Really Long String!

The Example

- Alphabet: $X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Rules:

$$r_0 : \; X \to F[+X][-X]FX \quad \big(\text{Replace all } X\text{'s by } F[+X][-X]FX\big)$$
$$r_1 : \; F \to FF \quad \big(\text{Replace all } F\text{'s by } FF\big)$$

- No rules for $[,], +, -$, so leave them alone.
- No need for $] \to]$ rules, etc.

Example Derivation

$$w_0 = X$$
$$w_1 = F[+X][-X]FX$$
$$w_2 = FF[+F[+X][-X]FX][-F[+X][-X]FX]FFF[+X][-X]FX$$
$$w_3 = \text{Really Long String!}$$

- Prohibitive to compute realistic strings by hand.

The Turtle Graphics Approach

- The Turtle is a drawing robot that sequentially interprets instruction string symbols in various ways:
  1. Moving forward (not important for us).
  2. Drawing a line segment as it moves forward.
  3. Turning (left or right) through a fixed angle.

The Turtle Graphics Approach

- The Turtle is a drawing robot that sequentially interprets instruction string symbols in various ways:
  1. Moving forward (not important for us).
  2. Drawing a line segment as it moves forward.
  3. Turning (left or right) through a fixed angle.
- And for branching structures:

The Turtle Graphics Approach

- The Turtle is a drawing robot that sequentially interprets instruction string symbols in various ways:
  1. Moving forward (not important for us).
  2. Drawing a line segment as it moves forward.
  3. Turning (left or right) through a fixed angle.
- And for branching structures:
  4. Remembering its current position.
  5. Returning to its most recently remembered position and then forgetting it.

The Turtle Graphics Approach

- The Turtle is a drawing robot that sequentially interprets instruction string symbols in various ways:
  1. Moving forward (not important for us).
  2. Drawing a line segment as it moves forward.
  3. Turning (left or right) through a fixed angle.
- And for branching structures:
  4. Remembering its current position.
  5. Returning to its most recently remembered position and then forgetting it.
- Prohibitive to plot realistic examples by hand.
- Small examples done by hand in practice classes to give insight into the process.
- Spreadsheets enable students to plot satisfying diagrams of branching structures.

The Turtle Graphics Approach

- The Turtle is a drawing robot that sequentially interprets instruction string symbols in various ways:
  1. Moving forward (not important for us).
  2. Drawing a line segment as it moves forward.
  3. Turning (left or right) through a fixed angle.
- And for branching structures:
  4. Remembering its current position (push).
  5. Returning to its most recently remembered position and then forgetting it (pop).
- Prohibitive to plot realistic examples by hand.
- Small examples done by hand in practice classes to give insight into the process.
- Spreadsheets enable students to plot satisfying diagrams of branching structures.
- Problem! Interpretations 4 and 5 require the use of a stack – difficult to implement in a spreadsheet.

The Turtle's World View

- At each time t, the turtle is at some location (x, y) and is pointing in direction α°.
- It has a fixed turn angle δ° and a movement unit d.

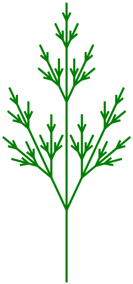
The Turtle's World View

- At each time t, the turtle is at some location (x, y) and is pointing in direction α°.
- It has a fixed turn angle δ° and a movement unit d.
- Instructions it understands:

Symbol	Interpretation	State Changes
F	move forward by d drawing a line	$x_{\text{new}} = x + d\cos(\alpha^\circ)$, $y_{\text{new}} = y + d\sin(\alpha^\circ)$
f	move forward by d without drawing a line	$x_{\text{new}} = x + d\cos(\alpha^\circ)$, $y_{\text{new}} = y + d\sin(\alpha^\circ)$
$+$	turn left by δ°	$\alpha^\circ_{\text{new}} = (\alpha + \delta)^\circ$
$-$	turn right by δ°	$\alpha^\circ_{\text{new}} = (\alpha - \delta)^\circ$
[	save current state	None
]	return to state saved at *previous* [	Revert
Other	Ignore	None

“Node rewrite” examples

Both $n = 5$.

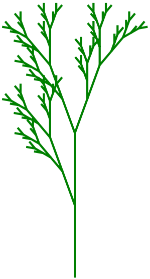


$\delta = 25°$

$w_0 = X$

$r_0 :\ X \to F[+X][-X]FX$

$r_1 :\ F \to FF$



$\delta = 20°$

$w_0 = X$

$r_0 :\ X \to F[+X]F[-X]+X$

$r_1 :\ F \to FF$

The Marvel of Growth

$$\delta = 25^\circ$$
$$w_0 = X$$
$$r_0 : \ X \to F[+X][-X]FX$$
$$r_1 : \ F \to FF$$

$n = 1$

The Marvel of Growth

$$\delta = 25^\circ$$
$$w_0 = X$$
$$r_0 : \; X \rightarrow F[+X][-X]FX$$
$$r_1 : \; F \rightarrow FF$$

$n = 1$

$n = 2$

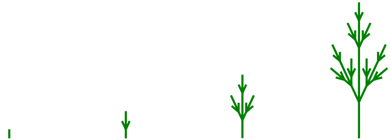
The Marvel of Growth

$$\delta = 25^\circ$$
$$w_0 = X$$
$$r_0 : \; X \rightarrow F[+X][-X]FX$$
$$r_1 : \; F \rightarrow FF$$

$n = 1$ $n = 2$ $n = 3$

The Marvel of Growth

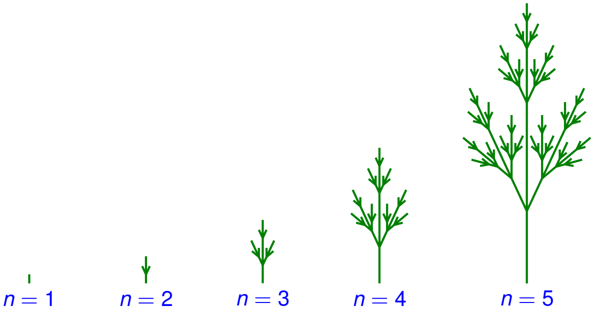
$$\delta = 25^\circ$$
$$w_0 = X$$
$$r_0 : \; X \to F[+X][-X]FX$$
$$r_1 : \; F \to FF$$



$n = 1$ $n = 2$ $n = 3$ $n = 4$

The Marvel of Growth

$$\delta = 25^\circ$$
$$w_0 = X$$
$$r_0 : \; X \to F[+X][-X]FX$$
$$r_1 : \; F \to FF$$



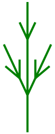
The Stack Bypassed

- We can sometimes avoid stacks using a modified Lindenmayer Grammar.
- The idea is to "retrace steps" back to the state where a [would have pushed the state.

The Stack Bypassed

- We can sometimes avoid stacks using a modified Lindenmayer Grammar.
- The idea is to "retrace steps" back to the state where a [would have pushed the state.
- To do this we introduce a new turtle instruction R meaning draw (move?) in reverse:

Symbol	Interpretation	State Changes
R	move backward drawing a line	$x_{\text{new}} = x + d\cos((180+\alpha)^\circ)$ $y_{\text{new}} = y + d\sin((180+\alpha)^\circ)$



Modifying the Example

- Alphabet: $R, X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Production Rules:

$$r_0 : \; X \to F + X - -X + FXRR$$
$$r_1 : \; F \to FF$$
$$r_2 : \; R \to RR$$

(No rules for $[,], +, -$, so leave them alone.)

- Example Derivation:
$w_0 = X$

Modifying the Example

- Alphabet: $R, X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Production Rules:

$$r_0 : \ X \to F + X - -X + FXRR$$
$$r_1 : \ F \to FF$$
$$r_2 : \ R \to RR$$

(No rules for $[,], +, -$, so leave them alone.)

- Example Derivation:
$w_0 = X$
$w_1 = F + X - -X + FXRR$

Modifying the Example

- Alphabet: $R, X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Production Rules:

$$r_0 : \; X \to F + X - -X + FXRR$$
$$r_1 : \; F \to FF$$
$$r_2 : \; R \to RR$$

(No rules for $[,], +, -$, so leave them alone.)

- Example Derivation:

$$w_0 = X$$
$$w_1 = F + X - -X + FXRR$$
$$w_2 = FF + F + X - -X + FXRR - -F + X - -X + FXRR + FFF -$$

Modifying the Example

- Alphabet: $R, X, F, [,], +, -$.
- Start word: $w_0 = X$.
- Production Rules:

$$r_0 : X \to F + X - -X + FXRR$$
$$r_1 : F \to FF$$
$$r_2 : R \to RR$$

(No rules for $[,], +, -$, so leave them alone.)

- Example Derivation:

$$w_0 = X$$
$$w_1 = F + X - -X + FXRR$$
$$w_2 = FF + F + X - -X + FXRR - -F + X - -X + FXRR + FFF$$
$$w_3 = \text{Ridiculously Long String!}$$

- It works!

The Question . . .

- This modified grammar approach raises an interesting mathematical problem:

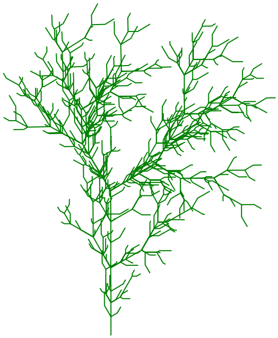
Question

Given an arbitrary Lindenmayer system, can we always find a modified grammar that generates the same diagrams?

If not, for what class of systems is this possible?

An “edge rewrite” system

- Not clear that modified grammar can always be found for “edge rewrite” systems. EG:



$$\delta = 30^\circ$$
$$w_0 = F$$
$$r_0 : \; F \to FF - [-F + F] + [+F - F]$$

References

- Aristid Lindenmayer, *Mathematical models for cellular interaction in development.*, J. Theoret. Biology, **18** 280-315, 1968.
- Przemyslaw Prusinkiewicz, *The Algorithmic Beauty of Plants* 1990,
  available online at: http://algorithmicbotany.org/