2-Day Introduction to Agent-Based Modelling

Day 2: Session 5

Variables and Debugging

2 dimensions of code context

Being aware of what the context of any command is important to understanding it.

- Which agent is doing the instruction: the observer, a kind of agent, a patch?
- In which procedure/ask is it?

Example:

  – ask sellers [set size 50]

2 dimensions of code context

Being aware of what the context of any command is important to understanding it.

- Which agent is doing the instruction: the observer, a kind of agent, a patch?
- In which procedure/ask is it?

Example:

  – ask sellers [set size 50]

This bit might be in the "setup" procedure in the observer context.

2 dimensions of code context

Being aware of what the context of any command is important to understanding it.

- Which agent is doing the instruction: the observer, a kind of agent, a patch?
- In which procedure/ask is it?

Example:

  – ask sellers [set size 50]

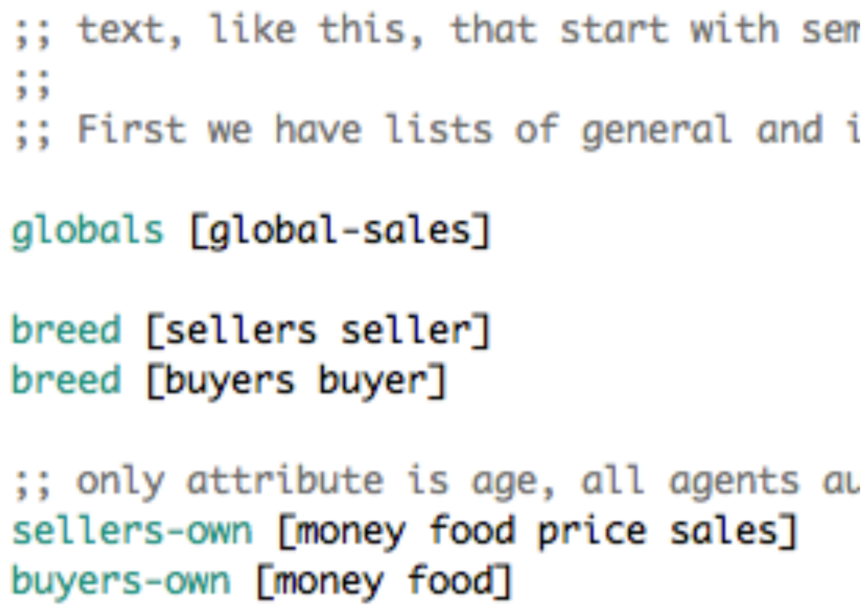
$\underbrace{\text{ask sellers [}}_{\text{This bit might be in the “setup” procedure in the observer context.}}\underbrace{\text{set size 50]}}_{\text{This bit is is in the seller context within the ask within the “setup” procedure.}}$

Variables with different scopes

- Variables are simply named "slots" that remember the values you set them to. You then use them by their name but they act like the value they are currently set to. E.g.
  - set price 4.30
  - set money money + 1
  - show [price] of sellers
- Different kinds of variable have different scopes – that is kinds of code context where they can be used by that name
- Variables listed in "global" or set in the interface (using a slider etc.) are global, and can be set or used anywhere
- Other variables are "local" to an agent such as color, xcor ycor, size and those listed in "…-own" in the code
- Another kind are local to a procedure, defined by "let" statements within the procedure
- When you use a variable that is local, which value you get is determined by the context of the statement

The “Market Simulation”

- Load the example simulation called “5-market-begin.nlogo”
- Look at the top of the code

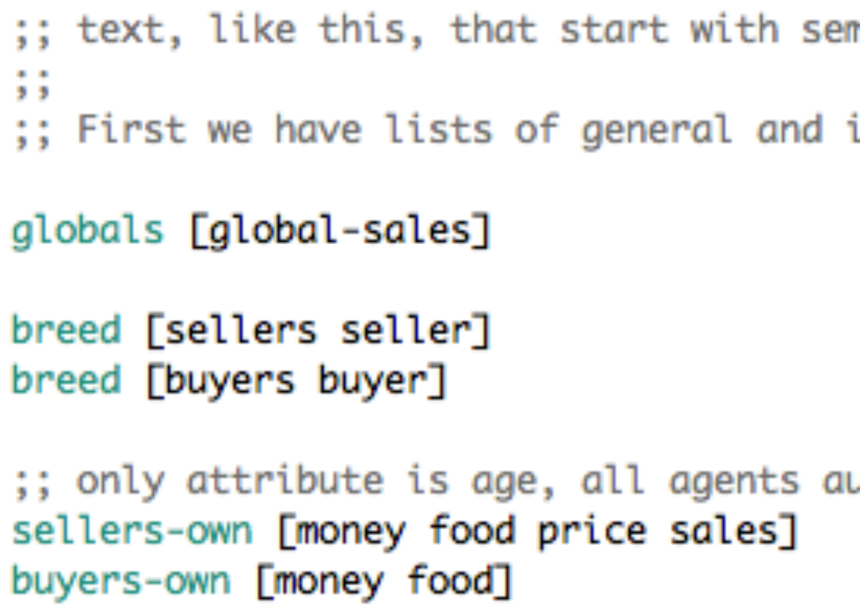


```
;; text, like this, that start with sem
;;
;; First we have lists of general and i

globals [global-sales]

breed [sellers seller]
breed [buyers buyer]

;; only attribute is age, all agents ar
sellers-own [money food price sales]
buyers-own [money food]
```

The “Market Simulation”

- Load the example simulation called “5-market-begin.nlogo”
- Look at the top of the code

This is a global variable, look for how it is used in the code to sum up all the sales of each seller. It is what is displayed by the “Tot.Sales” monitor in the Interface.



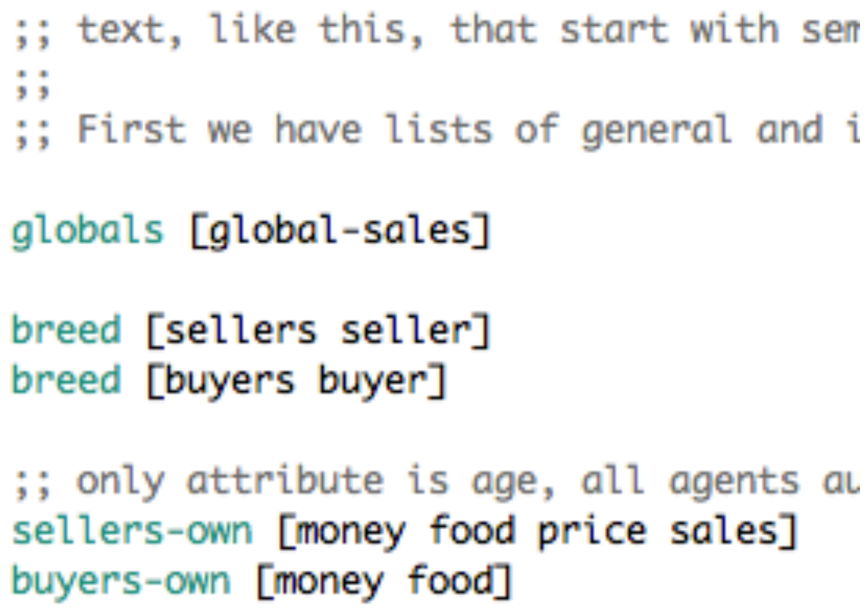
```
;; text, like this, that start with sem
;;
;; First we have lists of general and i

globals [global-sales]

breed [sellers seller]
breed [buyers buyer]

;; only attribute is age, all agents a
sellers-own [money food price sales]
buyers-own [money food]
```

The “Market Simulation”

- Load the example simulation called “5-market-begin.nlogo”
- Look at the top of the code

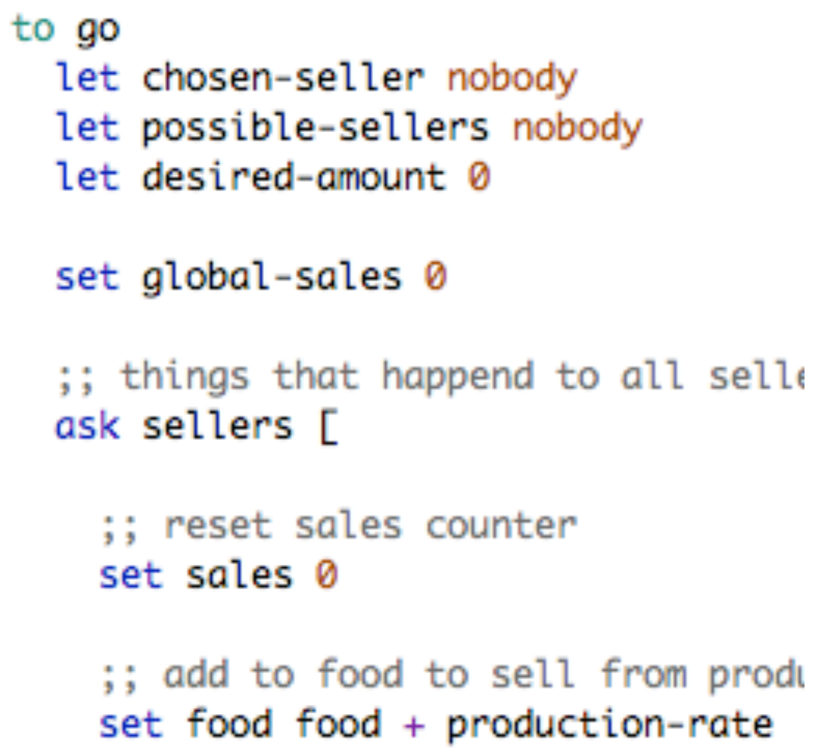
This is a global variable, look for how it is used in the code to sum up all the sales of each seller. It is what is displayed by the “Tot.Sales” monitor in the Interface.

Here the extra variables local to each seller and buyer is listed.



```
;; text, like this, that start with sem
;;
;; First we have lists of general and i

globals [global-sales]

breed [sellers seller]
breed [buyers buyer]

;; only attribute is age, all agents au
sellers-own [money food price sales]
buyers-own [money food]
```

Within a procedure

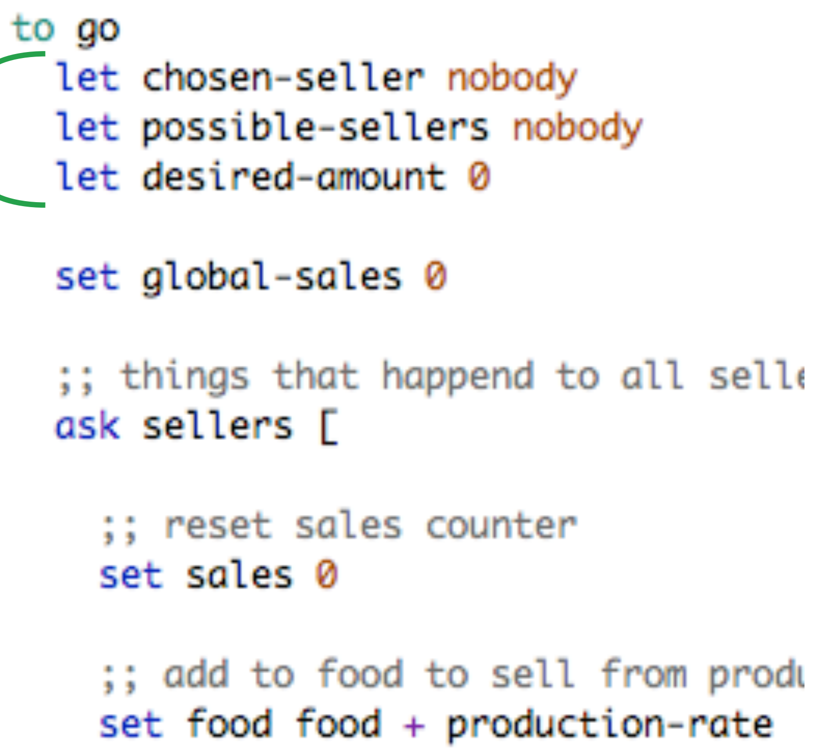


```
to go
  let chosen-seller nobody
  let possible-sellers nobody
  let desired-amount 0

  set global-sales 0

  ;; things that happend to all selle
  ask sellers [

    ;; reset sales counter
    set sales 0

    ;; add to food to sell from produ
    set food food + production-rate
```

Manchester Metropolitan University

Within a procedure

Here variables that are local to the "go" procedure are defined. They only have meaning within "go" or any procedure or agent called from "go".

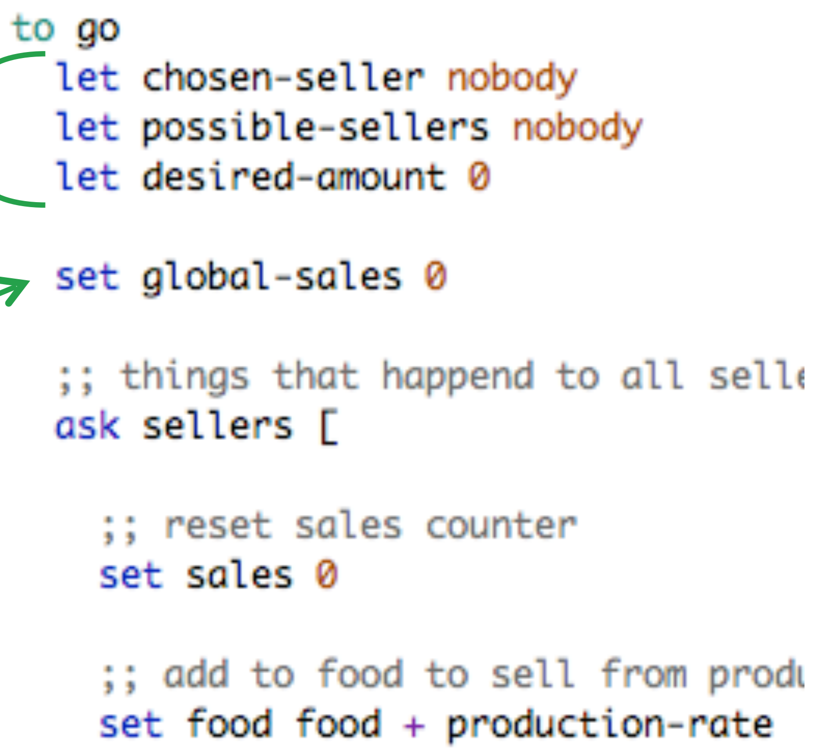


```
to go
  let chosen-seller nobody
  let possible-sellers nobody
  let desired-amount 0

  set global-sales 0

  ;; things that happend to all selle
  ask sellers [

    ;; reset sales counter
    set sales 0

    ;; add to food to sell from produ
    set food food + production-rate
```

Within a procedure

Here variables that are local to the "go" procedure are defined. They only have meaning within "go" or any procedure or agent called from "go".

Here the global variable is being set to zero at the start of a simulation time tick.



```
to go
  let chosen-seller nobody
  let possible-sellers nobody
  let desired-amount 0

  set global-sales 0

  ;; things that happend to all selle
  ask sellers [

    ;; reset sales counter
    set sales 0

    ;; add to food to sell from produ
    set food food + production-rate
```

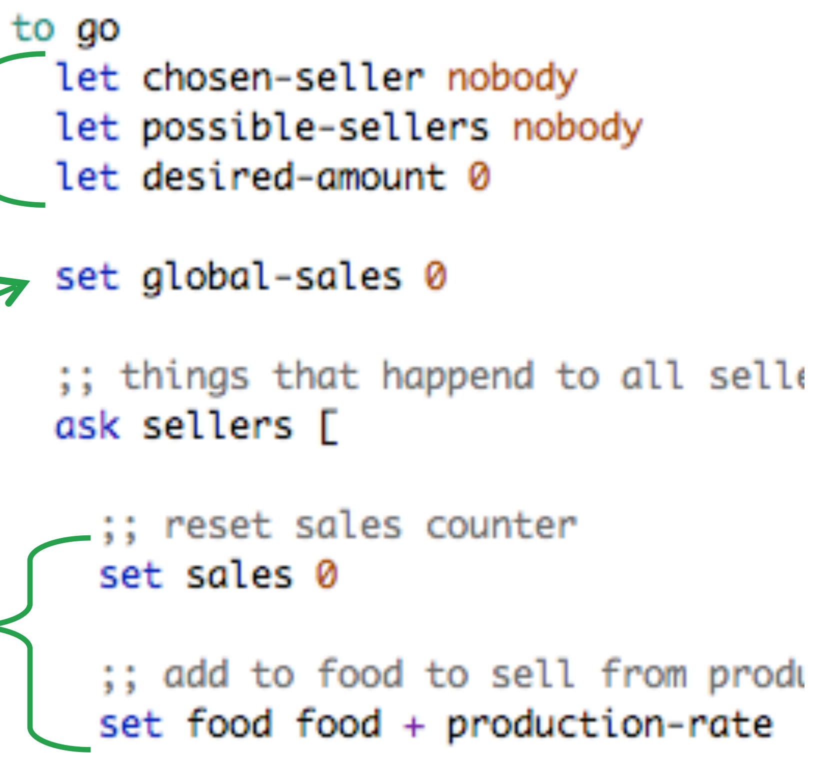
Manchester Metropolitan University

Within a procedure

Here variables that are local to the “go” procedure are defined. They only have meaning within “go” or any procedure or agent called from “go”.

Here the global variable is being set to zero at the start of a simulation time tick.

Variables local to each seller are being set and used here. Note “production-rate” can be used here because it is a global variable.



```
to go
  let chosen-seller nobody
  let possible-sellers nobody
  let desired-amount 0

  set global-sales 0

  ;; things that happend to all selle
  ask sellers [

    ;; reset sales counter
    set sales 0

    ;; add to food to sell from produ
    set food food + production-rate
```

Manchester Metropolitan University

About the Market Simulation

- There are buyers and sellers
- Buyers eat a certain amount of food each time tick, and have a given income
- Sellers have regular costs and production of food each time tick
- Buyers see if there are any sellers with a unit of food to sell, and if so look for the cheapest seller and buys what it can afford from this seller
- Sellers adjust their price depending on the level of their sales (dropping them if sales < 3 increasing them if > 6)

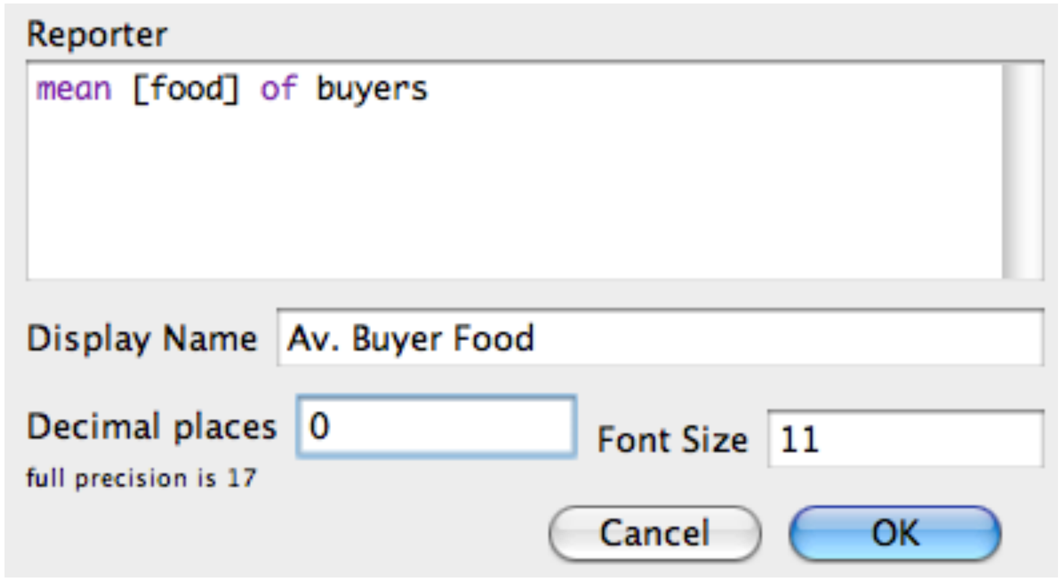
Manchester Metropolitan University

Play with the simulation

- Try different values for the global parameters set by the sliders
- Then step through the simulation
- Inspect different agents at different stages and see what state they are in (right-click on them select seller or buyer name and then inspect)
- Type in commands to query the state of the simulation, e.g.:
  - show [price] of sellers
  - show [food] of buyers
  - show mean [price] of sellers
  - show max [price] of sellers
  - Etc.
- Try playing with the simulation again looking at the values as they change in the simulation

Adding Monitors

- Right-click on some space and choose "Monitor"
- Fill in dialogue
  As shown here
  Then "**OK**"
- Maybe right-click on the Monitor that has now appeared and choose "select" then adjust its size and position
- Make other helpful monitors for money etc.



Reporter
```
mean [food] of buyers
```
Display Name: Av. Buyer Food

Decimal places: 0
full precision is 17

Font Size: 11

Manchester Metropolitan University

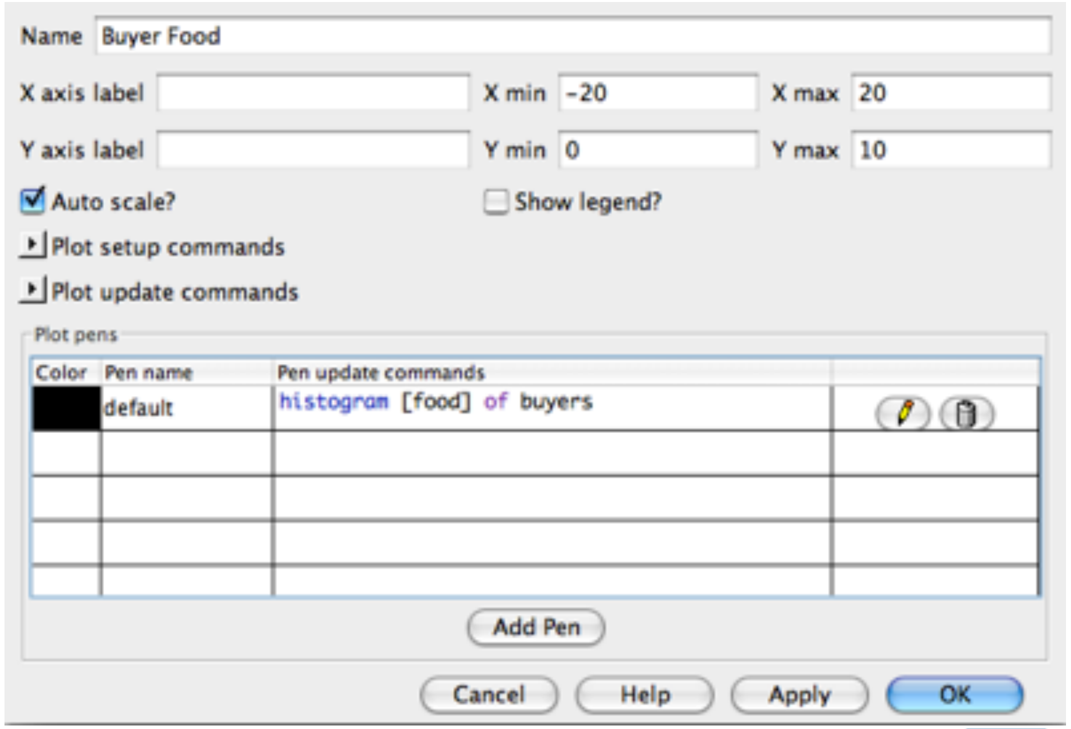
Adding graphs and histograms

Adding graphs and histograms

Expand the whole Interface window. Then right-click on some empty space and chose "plot". Then click on the small pencil icon in the plot dialogue and fill it in like this. Then "**OK**"

Adding graphs and histograms

Expand the whole Interface window. Then right-click on some empty space and chose "plot". Then click on the small pencil icon in the plot dialogue and fill it in like this. Then "**OK**"



Name: Buyer Food

X axis label		X min	-20	X max	20
Y axis label		Y min	0	Y max	10

[✓] Auto scale? [ ] Show legend?

▸ Plot setup commands

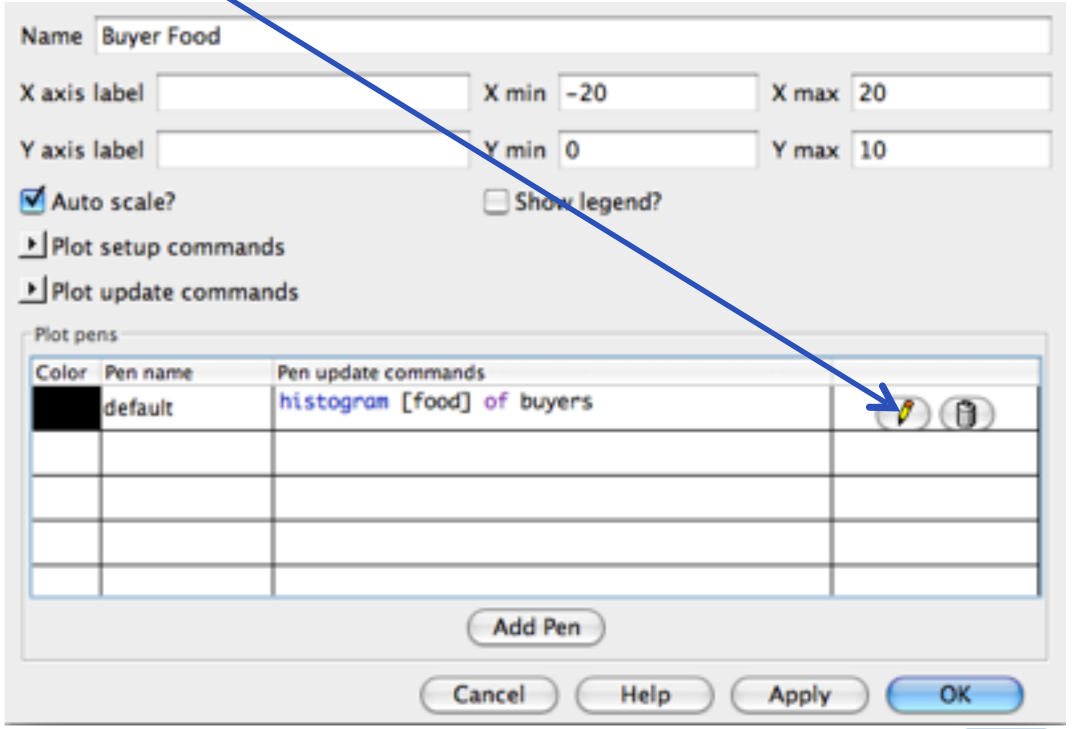
▸ Plot update commands

Plot pens

Color	Pen name	Pen update commands
■	default	histogram [food] of buyers

Adding graphs and histograms

Expand the whole Interface window. Then right-click on some empty space and chose "plot". Then click on the small pencil icon in the plot dialogue and fill it in like this. Then "**OK**"



Name	Buyer Food				
X axis label		X min	-20	X max	20
Y axis label		Y min	0	Y max	10

[✓ Auto scale?] Show legend? [ ]

Plot setup commands

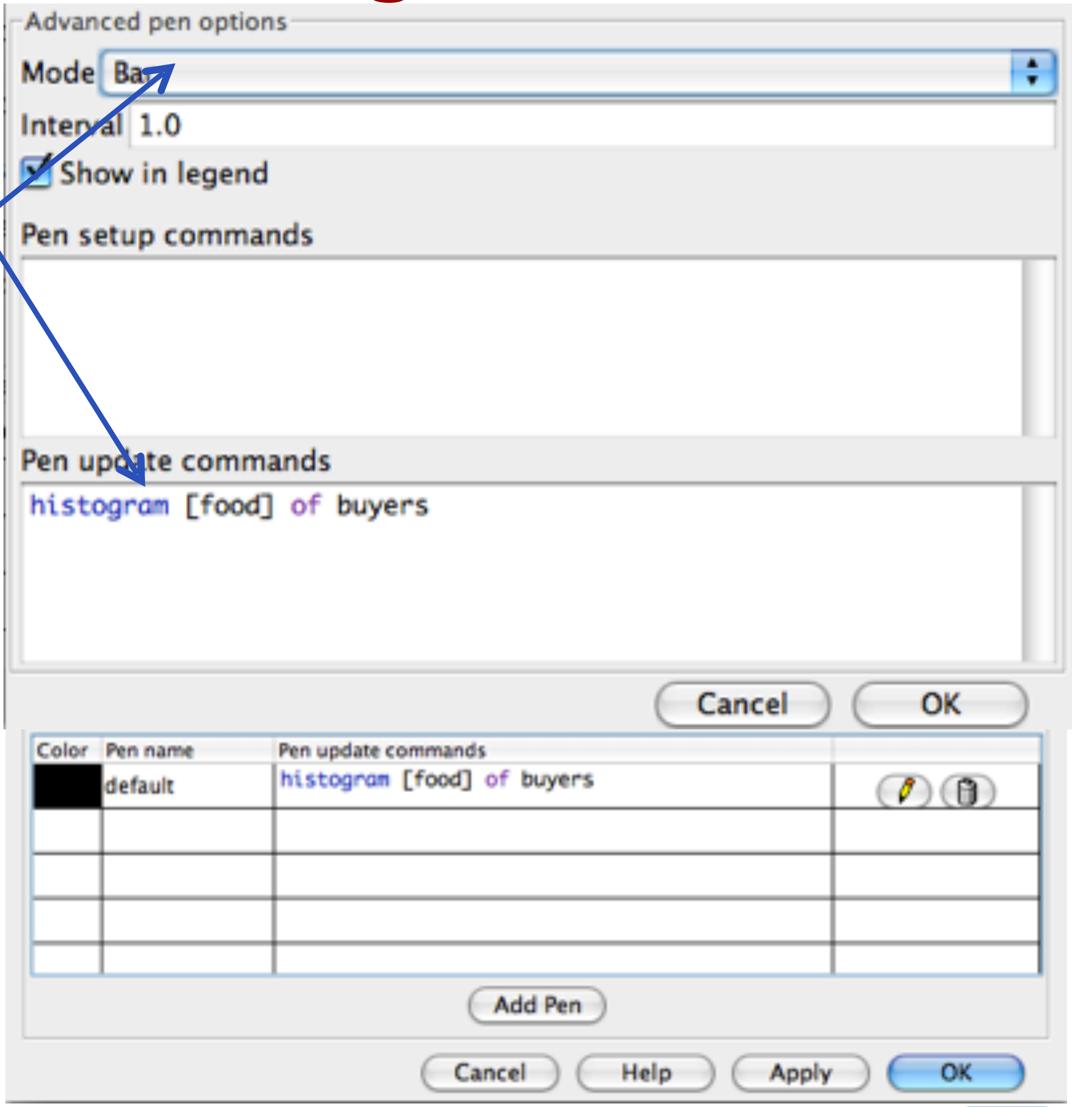
Plot update commands

Plot pens

Color	Pen name	Pen update commands
■	default	histogram [food] of buyers

Adding graphs and histograms

Expand the whole Interface window. Then right-click on some empty space and chose "plot". Then click on the small pencil icon in the plot dialogue and fill it in like this. Then "**OK**"



Advanced pen options

Mode: Bar

Interval: 1.0

☑ Show in legend

Pen setup commands

Pen update commands

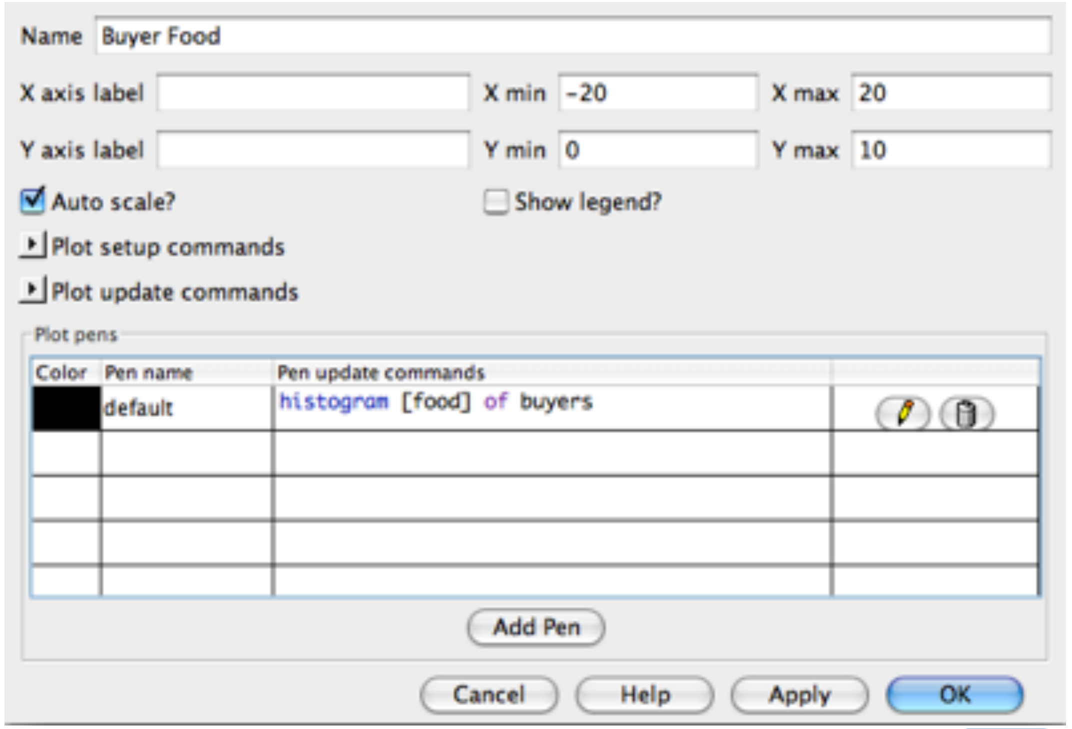
```
histogram [food] of buyers
```

Color	Pen name	Pen update commands
■	default	histogram [food] of buyers

Adding graphs and histograms

Expand the whole Interface window. Then right-click on some empty space and chose "plot". Then click on the small pencil icon in the plot dialogue and fill it in like this. Then "**OK**"

Then in the plot dialogue change the Name and the min and max values for the X axis. Then "**OK**"



Name	Buyer Food				
X axis label		X min	-20	X max	20
Y axis label		Y min	0	Y max	10

✓ Auto scale? ☐ Show legend?

▸ Plot setup commands

▸ Plot update commands

Plot pens

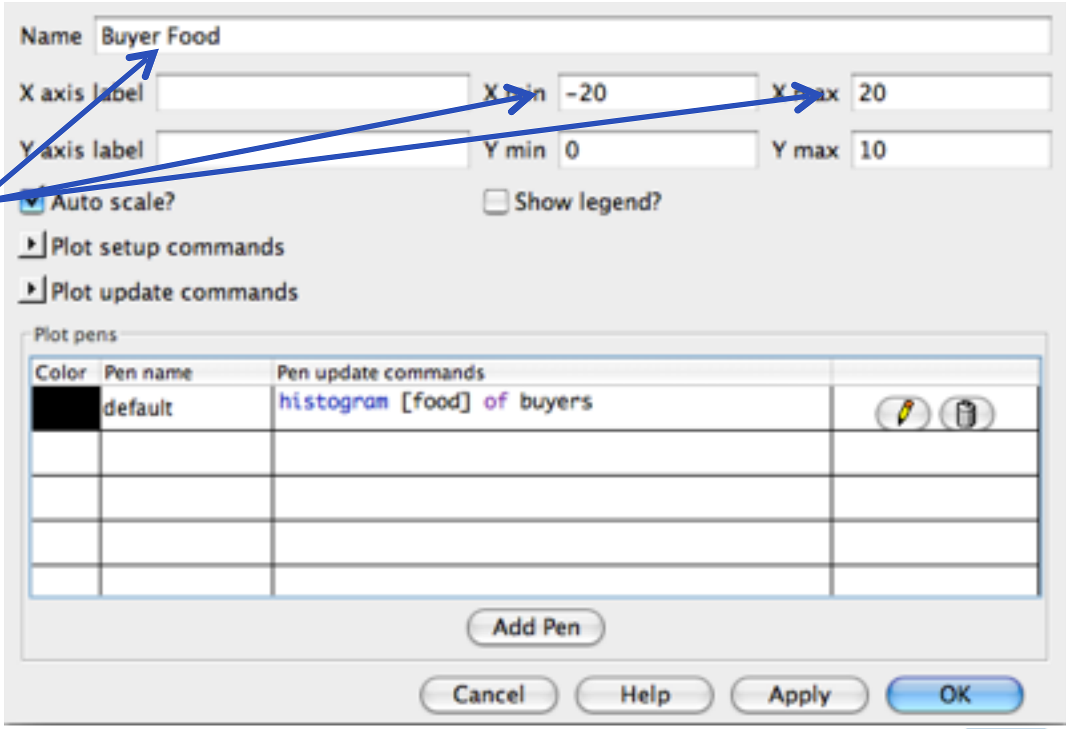
Color	Pen name	Pen update commands
■	default	`histogram [food] of buyers`

Manchester Metropolitan University

Adding graphs and histograms

Expand the whole Interface window. Then right-click on some empty space and chose "plot". Then click on the small pencil icon in the plot dialogue and fill it in like this. Then "**OK**"

Then in the plot dialogue change the Name and the min and max values for the X axis. Then "**OK**"



Name: Buyer Food

X axis label		X min	-20	X max	20
Y axis label		Y min	0	Y max	10

[✓] Auto scale?
[ ] Show legend?

Plot setup commands

Plot update commands

Plot pens

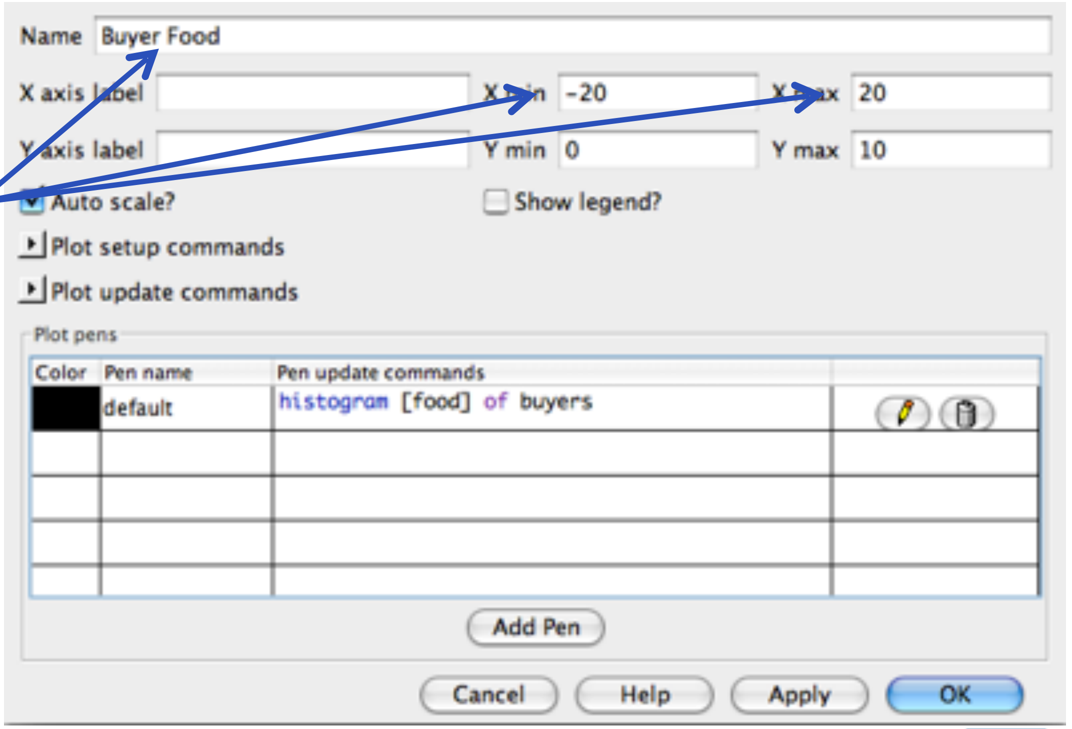
Color	Pen name	Pen update commands
■	default	`histogram [food] of buyers`

Adding graphs and histograms

Expand the whole Interface window. Then right-click on some empty space and chose "plot". Then click on the small pencil icon in the plot dialogue and fill it in like this. Then "**OK**"

Then in the plot dialogue change the Name and the min and max values for the X axis. Then "**OK**"

Go back to the simulation and play with it again, noticing the histogram of buyer food levels. Add some more histograms for: seller prices etc. you may have to go back and adjust ranges of X/Y axes.



Name: Buyer Food

X axis label		X min	-20	X max	20
Y axis label		Y min	0	Y max	10

[✓] Auto scale? ☐ Show legend?

▸ Plot setup commands

▸ Plot update commands

Plot pens

Color	Pen name	Pen update commands
■	default	histogram [food] of buyers

Some things to aim for…

- What additional aspects could you add to the simulation to make it more realistic?
- Can you change the simulation so that buyers/ sellers act more rationally?
- What happens if you “kill off” buyers with food < 0? (ask buyers with [food < 0] [die])
- What happens if you “kill off” sellers that go bankrupt (money < 0)?
- What behaviours lead to better/more stable market outcomes (i.e. without huge stocks of food, money, negative food etc.)?

Discussion – what were the main difficulties?

Stage of Debugging

- Most coding time is not spent making the original simulation but in “fixing” it or adding more code in (e.g. plots, new aspects)
- More careful design can help reduce time spent debugging but it still dominates

There are different stages of debugging (in order of both occurrence and difficulty):

1. Fixing syntax errors so the code runs
2. Making the micro-level behaviours work as you intended them to (verification)
3. Making the resultant outcomes be as you want them to be (tuning and validation)

Strategies for debugging

- Keep adding more graphs, monitors, visualisations etc. so you better understand what is happening
- Turn parts of the behaviours “off” by temporarily changing the code (e.g. make all prices fixed) to see what happens
- Temporarily adding “show” statements into the code to show what is happening, e.g.
  - `show (word "Food of " self " is " food)`

Agent Case Studies

- Global outputs such as graphs, monitors, visualisations can only go so far…
- There is often nothing else for it but to follow a particular agent step by step checking what it does and its state *each time* compared to the code
- This is time-consuming and everybody tries to avoid doing it…
- …you often will simply not really understand your simulation unless you do!

Manchester Metropolitan University

The End

2-Day Introduction to Agent-Based Modelling

http://cfpm.org/simulationcourse

Manchester Metropolitan University