

# Dealing with Data Gradients: “Backing Out” & Calibration

Nathaniel Osgood

MIT 15.879

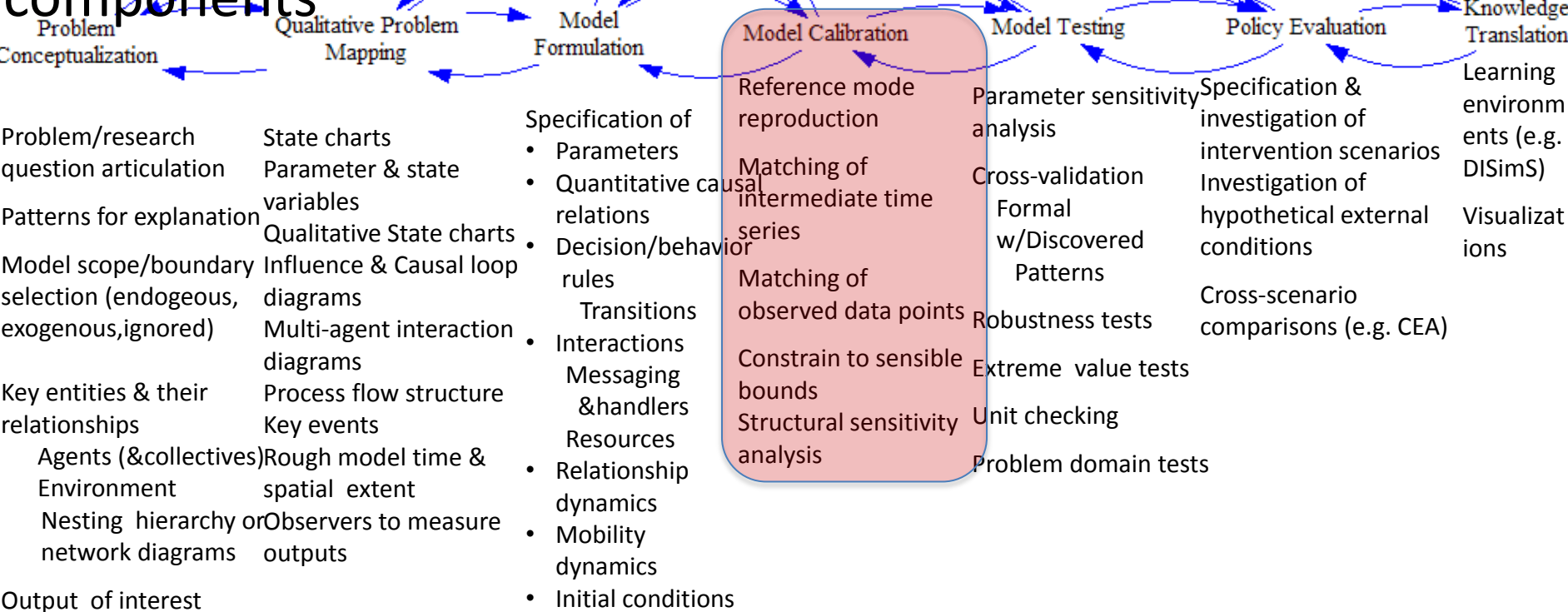
April 25, 2012

# ABM Modeling Process Overview

A Key Deliverable!

ODD:  
Design components  
& details

ODD: Overview  
& high-level design  
components



# Sources for Parameter Estimates

- Surveillance data
- Controlled trials
- Outbreak data
- Clinical reports data
- Intervention outcomes studies
- Calibration to historic data
- Expert judgement
- Metaanalyses

Parameter*	Description	Baseline value (units)	Reference
$\mu$	Entry/exit of sexual activity	0.0056 (years <sup>-1</sup> )	Garnett and Bowden, 2000
$c$	Partner change rate per Susceptible	16.08 (years <sup>-1</sup> )	Approximated from Garnett and Bowden, 2000
$\beta$	Probability of infection per sexual contact	0.70	Garnett and Bowden, 2000
$\phi$	Fraction of Infectives who are symptomatic	0.20	Garnett and Bowden, 2000
$1/\gamma$	Latent period	0.038 (years)	Brunham et. al., 2005
$1/\sigma$	Duration of infection	0.25 (years)	Brunham et. al., 2005
$\theta$	Asymptomatic recovery coefficient	1.5	Garnett and Bowden, 2000
$1/\pi$	Duration of naturally-acquired immunity	1 (year)	Approximated from Brunham et. al., 2005

# Sensitivity Analyses

- Same relative or absolute uncertainty in different parameters may have hugely different effect on outcomes or decisions
- Help identify parameters that strongly affect
  - Key model results
  - Choice between policies
- We place more emphasis in parameter estimation into parameters exhibiting high sensitivity

# Dealing with Data Gradients

- Often we don't have reliable information on *some* parameters, but do have other data
  - Often have data on emergent behavior of system – doesn't relate to any one parameter, but a combination influences
  - Some parameters may not be observable, but some closely related observable data is available
  - Sometimes the data doesn't have the detailed breakdown needed to specifically address one parameter
    - Available data could specify sum of a bunch of flows or stocks
    - Available data could specify some function of several quantities in the model (e.g. prevalence)
- Some parameters may implicitly capture a large set of factors not explicitly represented in model
- There are two big ways of dealing with this: manually “backing out”, and automated calibration



# “Backing Out”

- Sometimes we can manually take several aggregate pieces of data, and use them to collectively figure out what more detailed data might be
- Frequently this process involves imposing some (sometimes quite strong) assumptions
  - Combining data from different epidemiological contexts (national data used for provincial study)
  - Equilibrium assumptions (e.g. assumes stock is in equilibrium – deriving prevalence from incidence)
  - Independence of factors (e.g. two different risk factors convey independent risks)

# Example

- Suppose we seek to find out the sex-specific prevalence of diabetes in some population
- Suppose we know from published sources
  - The breakdown of the population by sex ( $c_M, c_F$ )
  - The population-wide prevalence of diabetes ( $p_T$ )
  - The prevalence rate ratio of diabetes in women when compared to men ( $rr_F$ )
- We can “back out” the sex-specific prevalence from these aggregate data ( $p_F, p_M$ )
- Here we can do this “backing out” without imposing assumptions



# Backing Out

# male diabetics + # female diabetics = # diabetics

$$(p_M * c_M) + (p_F * c_F) = p_T * (c_M + c_F)$$

- Further, we know that  $p_F / p_M = rr_F \Rightarrow p_F = p_M * rr_F$

- Thus

$$(p_M * c_M) + ((p_M * rr_F) * c_F) = p_T * (c_M + c_F)$$

$$p_M * (c_M + rr_F * c_F) = p_T * (c_M + c_F)$$

- Thus

- $p_M = p_T * (c_M + c_F) / (c_M + rr_F * c_F)$

- $p_F = p_M * rr_F = rr_F * p_T * (c_M + c_F) / (c_M + rr_F * c_F)$

# Disadvantages of “Backing Out”

- Backing out often involves questionable assumptions (independence, equilibrium, etc.)
- Sometimes a model is complex, with several related known pieces
  - Even though we may know a lot of pieces of information, it would be extremely complex (or involve too many assumptions) to try to back out several pieces simultaneously

# Another Example: Joint & Marginal Prevalence

	Rural	Urban	
Male	$p_{MR}$	$p_{MU}$	$p_M$
Female	$p_{FR}$	$p_{FU}$	$p_F$
	$p_R$	$p_U$	

Perhaps we know

- The count of people in each { Sex, Geographic } category
- Each marginal prevalence ( $p_R, p_U, p_M, p_F$ )

We need at least one more constraint (one possibility: assume  $p_{MR} / p_{MU} = p_R / p_U$ )

We can then derive the prevalence in each { Sex, Geographic } category

# Calibration: “Triangulating” from Diverse Data Sources

- Calibration involves “tuning” values of less well known parameters to best match observed data
  - Often try to match against *many* time series or pieces of data at once
  - Idea is trying to get the software to answer the question: “What must these (less known) parameters be in order to explain all these different sources of data I see”
- Observed data can correspond to complex combination of model variables, and exhibit “emergence”
- Frequently we learn from this that our model structure just can’t produce the patterns!

# Calibration

- Calibration helps us find a reasonable (specifics for) “dynamic hypothesis” that explains the observed data
  - Not necessarily the truth, but probably a reasonably good guess – at the least, a consistent guess
- Calibration helps us leverage the large amounts of diffuse information we may have at our disposal, but which cannot be used to directly parameterize the model
- Calibration helps us falsify models

# Calibration: A Bit of the How

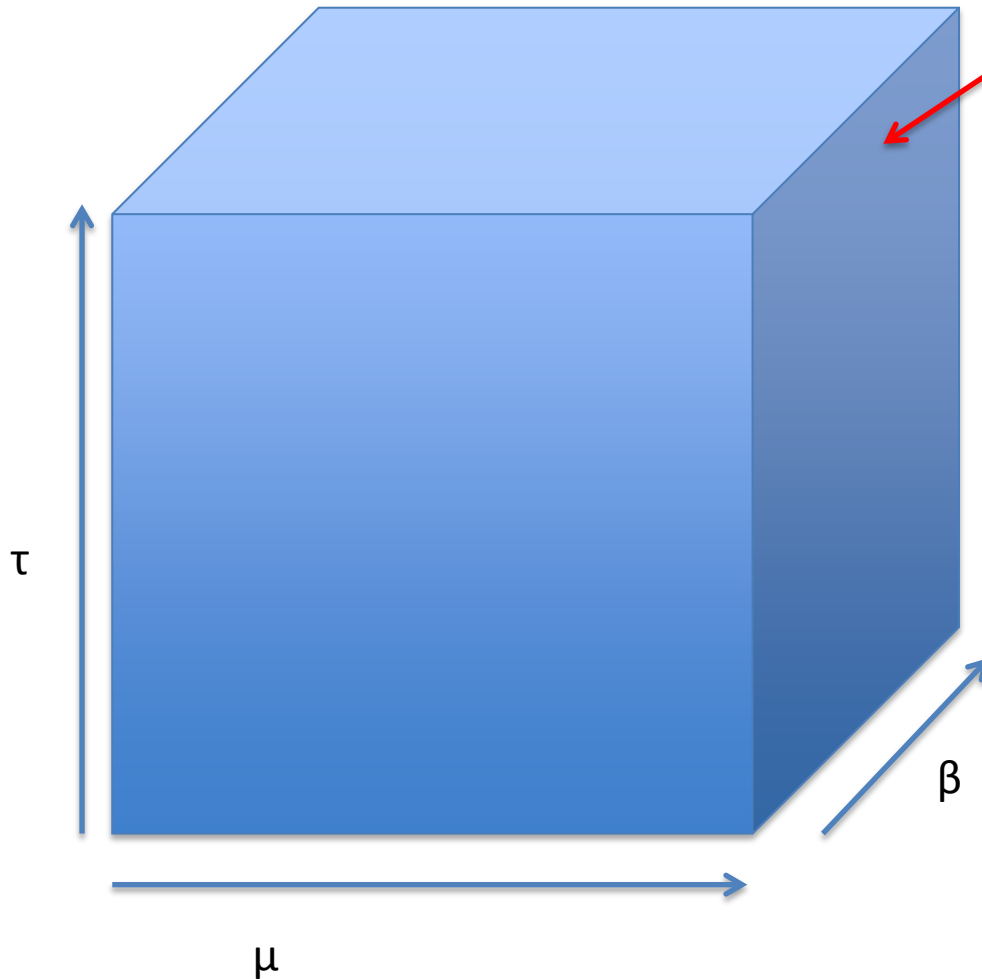
- Calibration uses a (global) optimization algorithm to try to adjust unknown parameters so that it automatically matches an arbitrarily large set of data
- The data (often in the form of time series) forms constraints on the calibration
- The optimization algorithm will run the model many (thousands or more) times to find the “best” match for all of the data

# Required Information for Calibration

- Specification of what to match (and how much to care about each attempted match)
  - Involves an “error function” ( “penalty function”, “energy function”) that specifies “how far off we are” for a given run (how good the fit is)
  - Alternative: specify “payoff function” (“objective function”)
- A statement of what parameters to vary, and over what range to vary them (the “parameter space”)
- Characteristics of desired optimization (tuning) algorithm
  - e.g. Single starting point of search?

# Envisioning “Parameter Space”

For each point in this space, there will be a certain “goodness of fit” of the model to the collective data





# Assessing Model “Goodness of Fit”

- To improve the “goodness of fit” of the model to observed data, we need to provide some way of quantifying it!
- Within the model, we
  - For each historic data, calculate discrepancy of model
    - Figure out absolute value of discrepancy from comparing
      - Historic Data
      - The model’s calculations
    - Convert the above to a fractional value (dividing by historic data)
  - Sum up these discrepancy

# Characteristics of a Desirable Discrepancy Metric

- **Dimensionless:** We wish to be able to add discrepancies together, regardless of the domain of origin of the data
- **Weighted:** Reflecting different pedigrees of data, we'd like to be able to weigh some matches more highly than others
- **Analytic:** We should be able to differentiate the function one or more times
- **Concave:** Two small discrepancies of size  $a$  should be considered more desirable than having one big discrepancy of size  $2a$  for one, and no discrepancy at all for the other.
- **Symmetric:** Being off by a factor of two should have the same weight regardless of whether we are  $2x$  or  $\frac{1}{2}x$
- **Non-negative:** No discrepancy should cancel out others!
- **Finite:** Finite inputs should yield finite discrepancies

# A Good Discrepancy Function (Assuming non-negative h & m)

Exponent  
>1  $\Rightarrow$  concave with respect to h-m

Taking average in denominator (together w/squaring of result) ensures symmetry with respect to h&m

$$w \cdot \left( \frac{h - m}{\text{average}(h, m)} \right)^2 = w \cdot \left( \frac{h - m}{\frac{h + m}{2}} \right)^2$$

Division  $\Rightarrow$  Dimensionless  
(Judging by *proportional* error, not absolute)

Only zero if h=m=0.

Denominator is only very small if numerator is as well!

# Considerations for Weighting

- **Purpose of model:** If we “care” more about a match with respect to some variables, we can more heavily weight matches for those variables
- **Uncertainty in estimate:** The more uncertain the estimate of the quantity, the lower the weight
- **Whether data exists:** no data => weight should be zero

# Example (Simplistic) Global Optimization Algorithm

- Starts at random position, tries to improve match (minimize error) by
    - Adjusting parameters
    - Running Model
    - Recording error function
  - Keeps on improving until reaches “local minimum” in error of fit
    - May add some randomness to knock out of local minima
- Many more sophisticated “global optimization” algorithms are available and can improve the outcome & speed of optimization (e.g. genetic algorithms, swarm-based methods)**



## Hands on Model Use Ahead



Load Sample Model:

**SIR Agent Based Calibration**

(Via “Sample Models” under “Help” Menu)

# An Optimization Experiment in AnyLogic

**Stops after best objective ceases to significantly improve**

**Caveat Modelor: May prematurely terminate the optimization**

**Stops after 500 optimization iterations**

**Varying these parameters**

**Calibration - Optimization Experiment**

Name: Calibration

Main active object class (root): Main

Random number generation:

- Random seed (unique simulation runs)
- Fixed seed (reproducible simulation runs)
- Custom generator (subclass of Random)

Objective:  minimize  maximize

difference( dsInfectiousCurrent, dsInfectiousHistoric )

Optimization stop conditions

- Iteration count: 500
- Automatic stop

Parameters:

Parameter	Type	Value		St
		Min	Max	
AverageI...uration	fixed	15		
ContactRate	continuous	0.5	5	0
Infection...bability	continuous	0.1	0.8	0
TotalPopulation	fixed	10000		

# An Optimization Experiment in AnyLogic Using Built-in Difference Function

The screenshot displays the AnyLogic software interface. On the left, a project tree shows a simulation named 'SIR Agent Based Calibration\*'. The main workspace shows the 'Calibration.java' file with the following code:

```
package sir_agent_based_calibration;

import java.sql.Connection;
import java.sql.SQLException;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Calendar;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.Currency;
import java.util.Date;
import java.util.Enumeration;
```

A blue text overlay reads: "A built-in objective function (euclidean distance)". A blue arrow points from this text to the 'difference' function in the 'Objective' field of the 'Calibration - Optimization Experiment' configuration window.

The configuration window shows the following settings:

- Name: Calibration
- Main active object class (root): Main
- Random number generation:  Random seed (Unique simulation runs)
- Objective:  minimize
- Objective function: `difference( dsInfectiousCurrent, dsInfectiousHistoric`

A tooltip for the 'difference' function is visible, showing its signature and description:

```
public static double difference(DataSet ds1,
                               DataSet ds2)
```

Difference function which is always not-negative and reflects difference between 2 given data sets in their common arguments range

**Parameters:**  
ds1 - data set  
ds2 - data set

**Returns:**  
square root of the average of square of difference between linearly interpolated data sets  
The integration range is the intersection of argument ranges of data sets



# Finding the Definition

The screenshot shows the AnyLogic University help interface. At the top, the search bar contains the text "difference dataset dataset" and the scope is set to "All topics". The search results are displayed in a list on the left, with the first result selected. The main content area on the right shows the definition for the "difference" function.

**Search Results**

- Collects data (PDF, CDF, etc.) of an array of histograms, each having a certain range of base (x) values and a range of data - y values. When an item (x,y) is added to Hist...**
- Compare Runs Experiment**  
This is an interactive experiment that allows you to input the model parameters, run simulation, and add the simulation output to the charts where they can be compared with...
- Calibration Experiment**  
When you have your model structure in place, you may wish to tune some parameters of the model so that its behavior in particular conditions matches a known (historical) pa...
- Sensitivity Analysis Experiment**  
This experiment helps you to explore how sensitive are the simulation results to changes of the model parameters. The experiment runs the model multiple times varying one o...
- Monte Carlo Experiment**  
Monte Carlo experiment obtains and displays a collection of simulation outputs for a stochastic model or for a model with stochastically varied parameter(s). You can find t...
- AnyLogic Professional**  
edition is the ultimate solution for development of large and complex simulation models and sophisticated animations, embedding models into various IT environments, and cre...
- Statistics**  
The Statistics object calculates statistical information (mean value, minimum, maximum, etc.) on a series of data samples of type double. The object works differently depen...
- AnyLogic 6.5 New Features**  
3D animation Easy access to MS Excel files on all platforms "How to..." models and other materials to support learning Model documentation in one click New objects and improv...
- Parameter Variation**  
AnyLogic affords an opportunity to run model with different model parameters and analyze how some certain parameters affect the model behavior. You don't need to run your m...
- Optimization Experiment**  
If you need to run a simulation and observe system behavior under certain conditions, as well as improve system performance, for example, by making decisions about system p...

**All Classes**

- [AbstractShapeGISMap](#)
- [ActiveObject](#)
- [ActiveObjectArrayList](#)
- [ActiveObjectCollection](#)
- [ActiveObjectIntegrationMan](#)
- [ActiveObjectLinkedHashSet](#)
- [ActiveObjectList](#)
- [Agent](#)
- [AgentContinuous](#)
- [AgentContinuous2D](#)
- [AgentContinuous3D](#)
- [AgentContinuousGIS](#)
- [AgentDiscrete2D](#)
- [Area2D](#)
- [Area3D](#)
- [BarChart](#)
- [Camera3D](#)
- [Chart](#)
- [Chart.Properties](#)
- [Chart1D](#)
- [Chart1DSum](#)
- [Chart2D](#)
- [Chart2DPlot](#)
- [Chart2DPlot.Appearance](#)
- [ChartItem](#)
- [Configuration3D](#)
- [CustomDistribution](#)
- [Database](#)
- [DataItem](#)
- [DataSet](#)

**difference**

```
public static double difference(DataSet ds1,
                               DataSet ds2)
```

Difference function which is always not-negative and reflects difference between 2 given data sets in their common arguments range

**Parameters:**

- ds1 - data set
- ds2 - data set

**Returns:**

- square root of the average of square of difference between linearly interpolated data sets
- The integration range is the intersection of argument ranges of data sets

**millisecond**

```
public double millisecond()
```

Returns a time value equal to one millisecond according to the current time unit setting.

**Returns:**

- a time value equal to one millisecond

**second**

```
public double second()
```

Returns a time value equal to one second according to the current time unit setting.

**Returns:**

- a time value equal to one second

**minute**

```
public double minute()
```

Returns a time value equal to one minute according to the current time unit setting.

# An Optimization Experiment in AnyLogic with a custom difference function

The screenshot displays the AnyLogic Advanced interface for an optimization experiment titled "Calibration of Agent Based SIR Model".

**Calibration Progress Graph:** The graph shows the "Calibration progress" on the y-axis (0 to 1) against an x-axis (0 to 0.7). A red line represents the current progress, which is currently "infeasible". A blue line indicates the "Custom distance function".

**Optimization Experiment Properties:**

- Objective:  minimize  maximize
- Function: `difference ()`
- Optimization stop conditions:
  - Iteration count: 500
  - Automatic Stop
- Parameters:

parameter	type	value	min	max	step
AverageL...uration	fixed	15			
ContactRate	continuous	0.5	0	3	0
Infection...bability	continuous	0.1	0	0.8	0
AreaSide	fixed	100			
TotalPopulation	fixed	10000			

# Defining a Payoff Function

## Caveat: Here, Non-Analytic, Non-Concave

AnyLogic Advanced [EDUCATIONAL USE ONLY]

File Edit View Model Window Help

Project Search

Parameters  
Plain Variables  
Embedded Objects  
Analysis Data  
Presentation  
Person  
Work  
Simulation: Main  
MonteCarlo1stOrder: Main  
SIR Agent Based Calibration\*  
Main  
Parameters  
Plain Variables  
Environments  
Embedded Objects  
Analysis Data  
dsInfectious  
Presentation  
Person  
Calibration: Main  
Functions  
InfectiousHistoric  
difference  
Analysis Data

Calibration

InfectiousHistoric

dsInfectiousHistoric

dsInfectiousCurrent

difference

dsInfectiousBest

Objective

Parameters

ContactRate ?

InfectionProbability ?

Copy the best solution to the clipboard

In this applet OptQuest optimizer i calibrate an agent based model o spread developed with AnyLogic. I each person is represented as a : (agent) with 4 possible states: Su Exposed, Infectious and Recovere Initially all but few people are susc few – exposed. A person can cont person, and in case one is susce another – exposed or infectious, t get infected with a certain probabi objective is to find the parameters (contact frequencies and infection so that the output of the simulator

Model

Parameter  
Flow Aux Variable  
Stock Variable  
Event  
Dynamic Event  
Plain Variable  
Collection Variable  
Function  
Table Function  
Port  
Connector  
Entry Point  
State  
Transition  
Initial State Pointer  
Branch  
History State  
Final State  
Environment

Problems

Description	Location
-------------	----------

Properties Console

difference - Function

General

Code

```
Function body:  
int diff = 0;  
for( int i=0; i<dsInfectiousCurrent.size(); i++ )  
    diff += abs( dsInfectiousCurrent.getY(i) - dsInfectiousHistoric.getY(i) );  
return diff;
```

**Computing absolute discrepancy  
Between historic & model values at  
specific point (index i) during realization**

# Historic Data Captured via Table Function

The screenshot displays the AnyLogic University interface for an SIR model calibration. The main window shows a simulation with a graph of 'Historic data, best fitting and current objective' over 10,000 iterations. The graph shows a red line for the best fit and a grey line for the current objective. The console window shows the following data:

Iteration:	Replication:	Objective:
	infeasible	infeasible
	?	?
	?	?

The 'Parameters' section shows:

Parameter	Value
ContactRate	?
InfectionProbability	?

The 'Table Function' configuration for 'InfectiousHistoric' is shown in the console window. The 'Interpolation' dropdown is set to 'Linear'. The 'Table Data' is as follows:

Argument	Value
2	3
4	8
6	24
8	71
10	202
12	558
14	1428
16	3070
18	5014
20	6214
22	6431
24	6083

A yellow arrow points from the text box to the 'Interpolation' dropdown menu.

How to  
interpolate  
("fill in")  
between data  
points

# Populating a Dataset with Historic Data

**Populating the dataset from the previously defined table function**

The screenshot displays the AnyLogic software interface. The main window shows a simulation environment with a graph and a table of results. The graph plots the 'Current objective' (blue line) and the 'Best fitting and current objective' (red line) over iterations. The table shows the following data:

Iteration:	?	?
Replication:	infeasible	infeasible
Objective:	?	?

The 'Advanced' tab of the 'Calibration - Optimization Experiment' properties window is active, showing the 'Initial experiment setup' section with the code:

```
dsInfectiousHistoric.fillFrom( InfectiousHistoric );
```

The 'Initial experiment setup' section also includes the following code:

```
datasetCurrentObjective.reset();  
datasetBestFeasibleObjective.reset();
```

# Stochastics in Agent-Based Models

- Recall that ABMs typically exhibit significant stochastics
  - Event timing within & outside of agents
  - Inter-agent interactions
- When calibrating an ABM, we wish to avoid attributing a good match to a particular set of parameter values simply due to chance
- To reliably assess fit of a given set of parameters, we need to repeatedly run model realizations
  - We can take the mean fit of these realizations

# Recall: Important Distinction (Declining Order of Aggregation)

- Experiment
  - Collection of simulations
- Simulation
  - Collection of replications that can yield findings across set of replications (e.g. mean value)
- Replication
  - One run of the model

# Populating the Appropriate Datasets

**Populates historic data up front from table fn**

**These datasets are within the experiment Persist beyond the simulation**

**If this is the best iteration, saves away the results**

**Retaining the Current value After the realization (Simulation run)**

```
dsInfectiousHistoric.fillFrom( InfectiousHistoric );
```

```
datasetCurrentObjective.reset();
```

```
datasetBestFeasibleObjective.reset();
```

```
dsInfectiousCurrent.fillFrom( root.dsInfectious );
```

```
if ( getCurrentIteration() == getBestIteration() )
```

```
    dsInfectiousBest.fillFrom( dsInfectiousCurrent );
```

Additional Class Code:

Initial Experiment Setup:

Before Each Experiment Run:

Before Simulation Run:

After Simulation Run:

After Iteration Code:

Run calibration

Current

Iteration: ?

Objective: ↓ ?

Parameters

Iteration: ?

InfectionProbability: ?

Best solution found

Model

- Parameter
- Flow Aux Variable
- Stock Variable
- Event
- Dynamic Event
- Plain Variable
- Collection Variable
- Function
- Table Function
- Port
- Connector
- Entry Point
- State
- Transition
- Initial State Pointer
- Branch
- History State
- Final State
- Environment

Action

Analysis

Presentation

Connectivity

Enterprise Library

More Libraries...



# Running Calibration in AnyLogic

**Calibration of Agent Based SIR Model**

Run calibration

technologies  
AnyLogic and this model is (c) XJ Technologies, www.anylogic.com. All rights reserved.

	Current	Best
Iteration:	5	3
Objective: ↓	120,500	3,895

**Parameters**

ContactRate	2.756	3
InfectionProbability	0.119	0.8

Copy the best solution to the clipboard

In this applet OptQuest optimizer is used to calibrate an agent based model of epidemic spread developed with AnyLogic. In that model each person is represented as a active object (agent) with 4 possible states: Susceptible, Exposed, Infectious and Recovered (SEIR). Initially all but few people are susceptible, and few – exposed. A person can contact another person, and in case one is susceptible and another – exposed or infectious, the first may get infected with a certain probability. The objective is to find the parameters of the agents (contact frequencies and infection probabilities) so that the output of the simulation model fits best with the historical data (in this case – the dynamics of infectious population). As the model is stochastic, the optimization is done under uncertainty, and simulation replications are used.

**Calibration progress**

Best payoff (objective) yet reached (lower is better)

**Historic data, best fitting and current simulation output**

Values of parameters being calibrated at best calibration thus far

Run: 4 Running Experiment: 1% Simulation: 5% 11.1 sec

# Optimization Constraints – Tests on Legitimacy of Parameter Values

**Calibration of Agent Based SIR Model**

Run calibration

Iteration: **infeasible** ?

Objective: ↓ ?

Parameters

Parameter	Current	Best
ContactRate	?	?

Calibration progress

Constraints on simulation parameters (are tested before a simulation run):

enabled	expression	type	bound
<input type="checkbox"/>			

Requirements (are tested after a simulation run to determine whether the solution is feasible):

enabled	expression	type	bound
<input type="checkbox"/>			

# Optimization Requirements – Tests to Sense Validity of Emergent Results

**Calibration of Agent Based SIR Model**

Run calibration

Iteration: **infeasible** ? ?

Objective: ↓ ? ?

**Parameters**

ContactRate ? ?

**Calibration progress**

1  
0.8  
0.6  
0.4

**Constraints on simulation parameters (are tested before a simulation run):**

enabled	expression	type	bound
<input type="checkbox"/>			

**Requirements (are tested after a simulation run to determine whether the solution is feasible):**

enabled	expression	type	bound
<input type="checkbox"/>			

# Enabling Multiple Realizations ("Replications", "Runs") per Iteration

The screenshot displays the AnyLogic Advanced software interface, specifically the Calibration - Optimization Experiment window. The interface is divided into several panes:

- Project Explorer (Left):** Shows a hierarchical tree of the model structure. Key elements include:
  - Parameters: InfectionProbability: 0.8, TotalPopulation: 10000
  - Plain Variables: nInfectious
  - Environments: Embedded Objects (people)
  - Analysis Data: dsInfectious
  - Presentation: people\_presentation
  - Person: Calibration: Main
    - Functions: InfectiousHistoric, difference
    - Analysis Data: datasetCurrentObjective, datasetBestFeasibleObjective, dsInfectiousHistoric, dsInfectiousCurrent, dsInfectiousBest
    - Presentation: MonteCarlo2DHistogram: Main
- Main Canvas (Center):** Displays the calibration experiment setup. A yellow callout box states: "These data correspond to ContactRate = 1.5, InfectionProbability = 0.4". The canvas contains several data sources:
  - datasetBestFeasibleObjective
  - InfectiousHistoric
  - dsInfectiousHistoric
  - dsInfectiousCurrent
  - difference
  - dsInfectiousBest
- Properties Panel (Right):** Shows the current state of the experiment:
  - Current:** Iteration: ? (infeasible), Objective: ?
  - Parameters:** ContactRate: ?, InfectionProbability: ?
  - Buttons: "Copy the best solution to the clipboard" and "In this applet OptQuest optimizer i calibrate an agent based model o spread developed with AnyLogic. I each person is represented as a : (agent) with 4 possible states: Su Exposed, Infectious and Recovere"
- Properties Panel (Bottom):** Shows the "Replications" section with the checkbox "Use replications" unchecked.
- Console (Bottom):** Displays the title "Calibration - Optimization Experiment".
- Palette (Far Right):** Lists various model components such as Parameter, Flow Aux Variable, Stock Variable, Event, Dynamic Event, Plain Variable, Collection Variable, Function, Table Function, Port, Connector, Entry Point, State, Transition, Initial State Pointer, Branch, History State, Final State, Environment, Action, Analysis, Presentation, Connectivity, and Enterprise Library.

# Fixed Number of Replications per Iteration

The screenshot displays the AnyLogic Advanced software interface. The main workspace shows a calibration model with various variables and functions. A red text overlay reads: "Specifies stopping Condition once minimum replications have been run. Indicates that the X% confidence interval around the mean is within 'Error percent' of the iteration mean obtained as of the most recent replication".

The Properties window is open to the "Replications" section, showing the following settings:

- Use replications
- Fixed number of replications
  - Replications per iteration: 10
- Varying number of replications (Stop replications after minimum replications, when confidence level is reached)
  - Minimum replications: 10
  - Maximum replications: 10
  - Confidence level: 80%
  - Error percent: 0.5

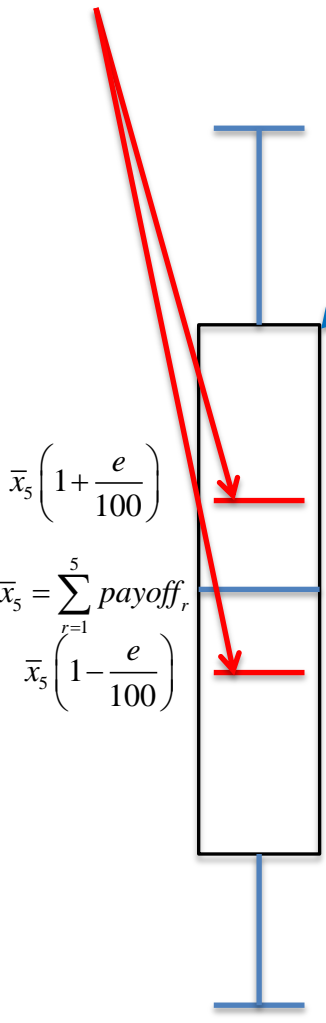
The "Error percent" field is circled in red, and a red arrow points from the text overlay to it.

Bars showing that delineating values within errorPercent% of mean

# Example

Terminates because confidence interval falls within errorPercent% bars

x% (e.g. 80%) confidence interval for sample mean (average) of replications to this point



$$\bar{x}_5 = \sum_{r=1}^5 \text{payoff}_r$$

$$\bar{x}_5 \left(1 + \frac{e}{100}\right)$$

$$\bar{x}_5 \left(1 - \frac{e}{100}\right)$$

After 5 replications

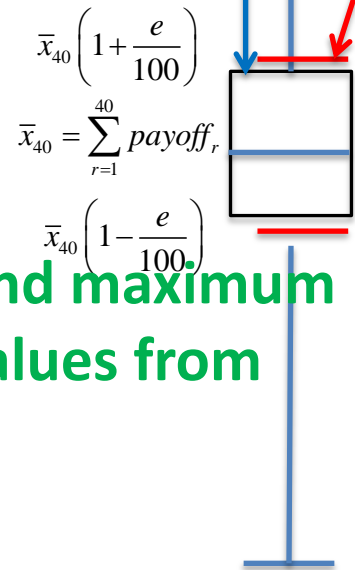


$$\bar{x}_{10} = \sum_{r=1}^{10} \text{payoff}_r$$

$$\bar{x}_{10} \left(1 + \frac{e}{100}\right)$$

$$\bar{x}_{10} \left(1 - \frac{e}{100}\right)$$

After 10 replications



$$\bar{x}_{40} = \sum_{r=1}^{40} \text{payoff}_r$$

$$\bar{x}_{40} \left(1 + \frac{e}{100}\right)$$

$$\bar{x}_{40} \left(1 - \frac{e}{100}\right)$$

After 40 replications  
**Terminates**

Minimum and maximum Observed values from replications

# Automatic Throttling of Replications Based on Empirical Fractiles for the Average of the Differences between Best and Current

The screenshot displays the AnyLogic Advanced software interface, specifically the 'Calibration - Optimization Experiment' window. The interface is divided into several panels:

- Project Explorer (Left):** Shows a hierarchical tree of model components, including 'InfectionProbability: 0.8', 'TotalPopulation: 10000', 'Plain Variables', 'nInfectious', 'Environments', 'Embedded Objects', 'people', 'Analysis Data', 'dsInfectious', 'Presentation', 'people\_presentation', 'Person', 'Calibration: Main', 'Functions', 'InfectiousHistoric', 'difference', 'Analysis Data', 'datasetCurrentObjective', 'datasetBestFeasibleObjective', 'dsInfectiousHistoric', 'dsInfectiousCurrent', 'dsInfectiousBest', 'Presentation', and 'MonteCarlo2DHistogram: Main'.
- Main Canvas (Center):** Displays a grid with several data points and a text box. The text box contains: "These data correspond to ContactRate = 1.5 InfectionProbability = 0.4". Below the text box, there are several data points labeled 'InfectiousHistoric', 'dsInfectiousHistoric', 'dsInfectiousCurrent', 'difference', and 'dsInfectiousBest'.
- Properties Panel (Bottom Left):** Shows the 'Replications' section with the following settings:
  - Use replications
  - Fixed number of replications
  - Replications per iteration: 10
  - Varying number of replications (Stop replications after minimum replications, when confidence level is reached)
  - Minimum replications: 10
  - Maximum replications: 100
  - Confidence level: 80%
  - Error percent: 0.5
- Console (Bottom Center):** Displays the title 'Calibration - Optimization Experiment'.
- Right Panel (Right):** Contains a 'Palette' with various model components and a 'Current' section showing 'Iteration: ?' and 'Objective: ?' (marked as 'infeasible'). Below this is a 'Parameters' section with 'ContactRate' and 'InfectionProbability' (both marked as '?'). A yellow vertical bar is visible next to the parameters. At the bottom of the right panel, there is a 'More Libraries...' button.

# Enabling Random Variation Between Realizations (“Replications”)

The screenshot displays the AnyLogic Advanced software interface, specifically the Calibration - Optimization Experiment window. The main workspace shows a grid with several data points and a text box indicating: "These data correspond to ContactRate = 1.5, InfectionProbability = 0.4". A yellow callout box highlights the "InfectiousHistoric" data point.

The right-hand side of the interface features a "Current" section with the following values:

Iteration:	Current
Iteration:	infeasible
Objective:	↓

Below this is the "Parameters" section:

Parameter	Value
ContactRate	?
InfectionProbability	?

The bottom section of the interface is the "Calibration - Optimization Experiment" configuration panel:

- General:** Name: Calibration; Main active object class (root): Main; Ignore: ; Create Def:
- Advanced:** Random number generation:  Random seed (unique simulation runs);  Fixed seed (reproducible simulation runs) Seed Value: 1
- Objective:**  minimize;  maximize; Objective function: difference()
- Optimization stop conditions:**  Iteration count: 500;  Automatic Stop
- Parameters:** (Empty list)

The left-hand side shows a Project tree with various model components like InfectionProbability, TotalPopulation, and dsInfectious. The bottom-left corner contains a Problems table with columns for Description and Location.



# Understanding Replications: Report Results for Each Replication!

The screenshot displays the AnyLogic Advanced software interface, specifically the Calibration - Optimization Experiment window. The interface is divided into several panels:

- Project Panel (Left):** Shows a hierarchical tree of model components. Under "Calibration: Main", the "Functions" folder contains "InfectiousHistoric", "difference", and "dsInfectiousCurrent". The "Analysis Data" folder contains "datasetCurrentObjective", "datasetBestFeasibleObjective", "dsInfectiousHistoric", "dsInfectiousCurrent", and "dsInfectiousBest".
- Main Canvas:** A grid-based workspace containing several data sets (ds) and functions. A yellow tooltip box is visible, stating: "These data correspond to ContactRate = 1.5, InfectionProbability = 0.4".
- Properties Panel (Right):** Shows the "Current" state of the calibration. It indicates "Iteration: ?" and "Objective: ?" with a red "infeasible" status. A "Parameters" section lists "ContactRate" and "InfectionProbability". A button "Copy the best solution to the clipboard" is present.
- Console Panel (Bottom):** Displays the "Calibration - Optimization Experiment" configuration. It includes sections for "Before Each Experiment Run", "Before Simulation Run", "After Simulation Run", and "After Iteration Code".

```
dsInfectiousHistoric.fillFrom( InfectiousHistoric );

Before Each Experiment Run:
datasetCurrentObjective.reset();
datasetBestFeasibleObjective.reset();

Before Simulation Run:

After Simulation Run:
dsInfectiousCurrent.fillFrom( root.dsInfectious );
println("For this particular simulation, the difference is\t" + difference());

After Iteration Code:
if( getCurrentIteration() == getBestIteration() )
    dsInfectiousBest.fillFrom( dsInfectiousCurrent );
```

The console also shows a "Tolerance:" label at the bottom.

# During First Several Realizations (“Replications”, “Runs”), No Results Appear

**Calibration of Agent Based SIR Model**

Run calibration

**Current Best**

Iteration:	1
Objective: ↓	3,343.5

**Parameters**

ContactRate 1.75

InfectionProbability 0.45

Copy the best solution to the clipboard

**Calibration progress**

Legend: ■ Current objective, — Best feasible objective

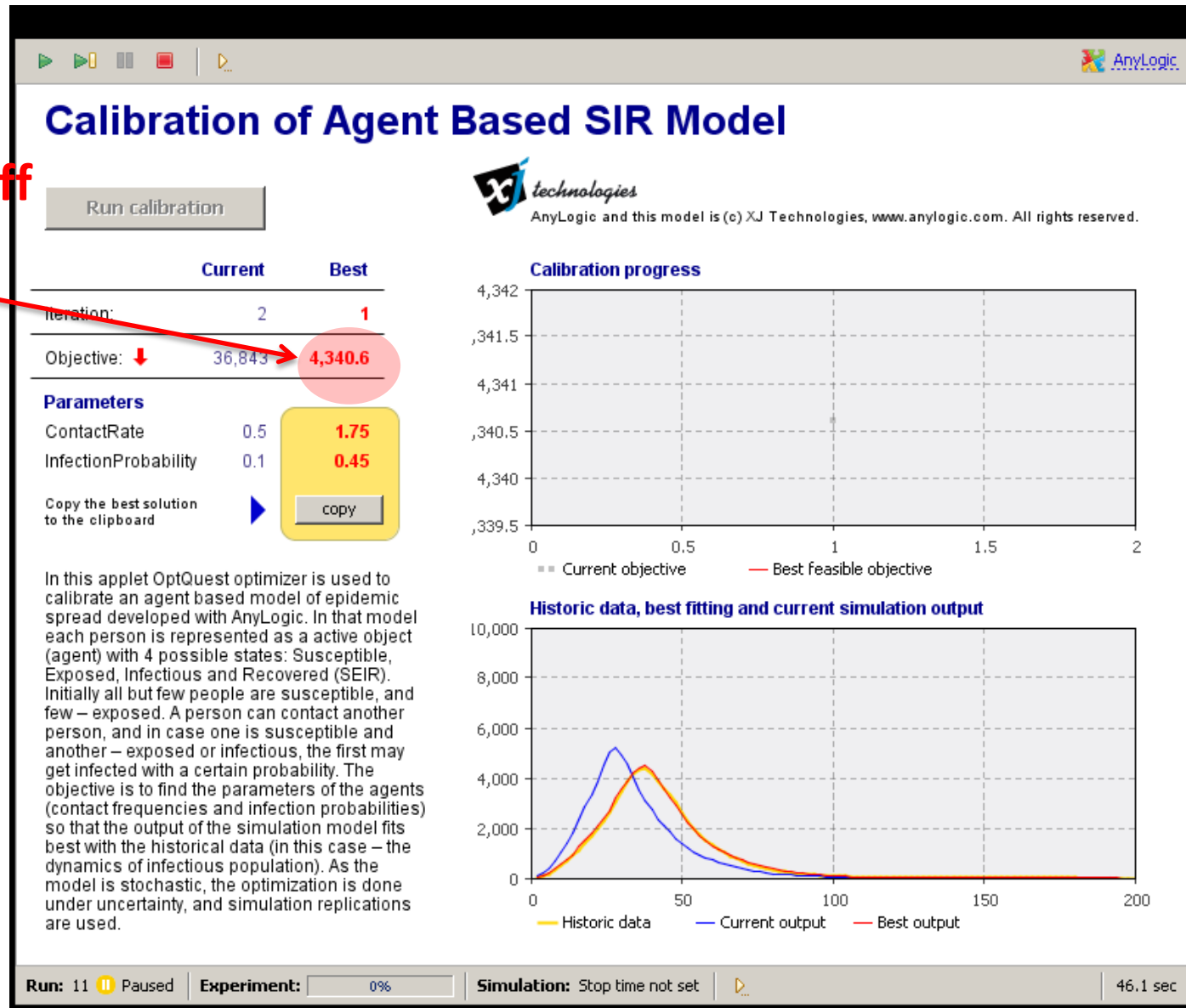
**Historic data, best fitting and current simulation output**

Legend: — Historic data, — Current output, — Best output

**Run:** 2 Running **Experiment:** 0% **Simulation:** 21% 4.6 sec

# Report on Iteration 1 Appears after a Count of Runs Equal to Replications per Iteration

Reports best payoff (objective) yet reached (lower is better), but from where did this number come?



# Output

The screenshot displays the AnyLogic Advanced interface. The main workspace shows a grid with several data series: datasetBestFeasibleObjective, InfectiousHistoric, dsInfectiousHistoric, dsInfectiousCurrent, difference, and dsInfectiousBest. A text box in the workspace states: "These data correspond to ContactRate = 1.5, InfectionProbability = 0.4". The right-hand side features a "Current" table with "Iteration:" and "Objective:" (marked as infeasible), and a "Parameters" section for ContactRate and InfectionProbability. Below the workspace is a console window showing the following output:

```
anylogic config [Java Application] C:\Program Files (x86)\AnyLogic 6\jre\bin\javaw.exe (Mar 31, 2011 4:44:06 AM)
For this particular simulation, the difference is 3324
For this particular simulation, the difference is 3363
For this particular simulation, the difference is 8866
For this particular simulation, the difference is 2052
For this particular simulation, the difference is 2447
For this particular simulation, the difference is 3552
For this particular simulation, the difference is 6079
For this particular simulation, the difference is 6082
For this particular simulation, the difference is 4775
For this particular simulation, the difference is 2866
For this particular simulation, the difference is 36843
```

A red callout box highlights the values 3324, 3363, 8866, 2052, 2447, 3552, 6079, 6082, 4775, and 2866, with a red arrow pointing to the final value 36843. A red text overlay at the bottom reads: "The reported payoff for the iteration is the average of the payoffs for each replication within the replication".

# Average of Results for Replications is the Reported Score for the Iteration!

The screenshot shows the Microsoft Excel interface with the following data and formula:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1															
2															
3			For this pa	3324											
4			For this pa	3363											
5			For this pa	8866											
6			For this pa	2052											
7			For this pa	2447											
8			For this pa	3552											
9			For this pa	6079											
10			For this pa	6082											
11			For this pa	4775											
12			For this pa	2866											
13															
14				4340.6											
15															
16															
17															
18															
19															
20															
21															
22															
23															
24															

The formula bar shows the formula: `=AVERAGE(D3:D12)`

# Considerations

- Adding constraints helps increase identifiability (selection of realistic best fit)
- Adding parameters to tune leads to larger space to explore
- Adding too many parameters to tune can lead to underdetermined situation
- All fits are within constraints of model

# Dealing with Calibration Problems: Experiments

- Try to “outsmart” calibration
  - Adopt best parameter values from calibration
  - Try to adjust parameters to do better than calibration
    - If is better, it may be that the parameter space is too large, or that the range constraints are too tight
    - Typically this does not do as well: Opportunity to learn
      - Model not respond in the way that anticipated to parameter change
      - May just shift the discrepancy from one variable to another
        - » Assumptions of model structure/values may not permit both variables to simultaneously match well!
- Set very high weight on thing that want to match, and see other matches
- Set all other weights to 0 (see if can possibly match)

# Dealing with Calibration Problems: Additional Experiments

- Increase parameter range
- Increase # of parameters
- Examine impact of changed model structure
- Run for larger number of optimization runs
- Find other estimates for uncertain parameters



# Important Cross-Checks: Uniqueness

- Are the calibration values Unique? If so, good; if not,
  - Do they give the same underlying interpretation?
  - Do the different interpretations lead to parameters that “trade off” in some structured way?
- Ways of addressing significantly different interpretations
  - Collect more primary data!
  - Impose additional constraints (in terms of time series, etc.)
  - Simplify model
  - Find other estimates for uncertain parameters

# Important Cross-Checks: Binding Constants

- Look for calibrated parameter values that are at the edges of their permissible ranges
  - If “best” value is at the edge of the range, it may be that even better calibrations would have been possible if continuing in that direction
- To deal with those at the edge
  - Relax constraints
  - Collect more data on plausible values
  - Question model structure

# Capturing Parameter Interdependencies in Calibration

- If we want parameter B adjusted during calibration to be at least as big as parameter A
  - In vensim, we can't enforce this constraint using the typical calibration machinery, because the range limits for parameters must be constants
  - we can accomplish this by calibrating only parameter A, and a parameter representing the ratio B/A.
- If we want to adjust two or more parameters such that they still sum to 1 (e.g. fraction of initial population in each of  $n$  or more stocks), we can adjust each of  $n$  non-normalized weights, and then take the corresponding normalized amount to be frac. falling in that category

# Calibrating Initial Conditions

- The initial conditions can be one of the best values to calibrate
- Sometimes need to divide a fixed population into several stocks

# Calibration & Regression: Similarities & Differences

- Model calibration is similar to regression in that we are seeking to find the parameter values allowing the best match of model & data
  - As in non-linear regression, for non-linear simulation models no “closed form” solution of best parameter values is possible  $\Rightarrow$  optimization is required
- A big difference:
  - **Regression models:** the “functional form” (dependence of model output on par’ms/indep vars) is given explicitly
  - **Simulation models:** behavior is only *implicitly* specified (e.g. via giving differentials); model output is a complex resultant (even emergent) property of structure