

Recall: 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*' with a 'Main' object containing a 'Calibration' sub-object. 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 'Objective' field in the 'Calibration - Optimization Experiment' configuration window. In this window, the 'Objective' is set to 'minimize' and the function is 'difference(dsInfectiousCurrent, dsInfectiousHistoric)'. A tooltip for the 'difference' function is visible, providing details about its parameters and return value.

Calibration - Optimization Experiment

Name: Calibration Main active object class (root): Main Ignore
 Create default UI

Random number generation:
 Random seed (Unique simulation runs)
 Fixed seed (reproducible simulation runs) Seed value: 1
 Custom generator (subclass of Random): new Random()

Objective: minimize maximize
 difference(dsInfectiousCurrent, dsInfectiousHistoric)

Optimization stop:
 Iteration count
 Automatic stop

Parameters:
 Parameter: AverageI...uratic

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

Calibration of Agent Based SIR Model

Run calibration

	Current	Best
Iteration:	?	?
Objective: ↓	?	?

Parameters

ContactRate	?	?
InfectionProbability	?	?

Copy the best solution to the clipboard

copy

Calibration progress

Custom distance function

Properties Console

Calibration - Optimization Experiment

Objective: minimize maximize

difference ()

Optimization stop conditions

Iteration count: 500

Automatic Stop

Parameters:

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

Varying these parameters

Selection

Defining a Payoff Function

Caveat: Here, Non-Analytic, Non-Concave

The screenshot shows the AnyLogic Advanced software interface. The main workspace displays a calibration model for an SIR agent-based model. The model includes variables such as `dsInfectiousHistoric`, `dsInfectiousCurrent`, `difference`, and `dsInfectiousBest`. The `difference` function is defined in the console window, and its code is shown in the code editor below. The code calculates the absolute discrepancy between historic and model values at a specific point (index `i`) during realization.

```
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 software interface during a calibration process. The main window shows a simulation of an SIR model with a graph of the objective function. The graph shows a red line representing the best fit and a grey line representing the current objective. The x-axis is labeled 'f' and ranges from 0 to 0.4. The y-axis ranges from 0.3 to 0.8. The graph shows a sharp peak at approximately x=0.1, followed by a decline and then a slight increase. The text 'Historic data, best fitting and curr' is visible at the bottom of the graph.

The console window shows the following text:

```
Iteration: ?  
Replication: infeasible ? infeasible ?  
Objective: ↓ ? ?  
Parameters  
ContactRate ?  
InfectionProbability ?  
Copy the best solution to the clipboard [copy]  
The built-in OptQuest optimizer is used to calibrate  
a compartment-based model of infectious disease diffusion.
```

The table function configuration panel for 'InfectiousHistoric' is shown below:

General

Name: InfectiousHistoric Show name Ignore Show at runtime

Access: public Static

Interpolation: Linear

Out of Range: Error

Table Data:

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, which is currently set to 'Linear'.

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 for a calibration experiment. The main workspace shows a table of parameters and a graph of objectives. The 'Advanced' tab of the 'Calibration - Optimization Experiment' properties window is active, showing the 'Initial experiment setup' section with the code `dsInfectiousHistoric.fillFrom(InfectiousHistoric);`. A red arrow points from a text box on the left to this code line.

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

Parameters

ContactRate	?	?
InfectionProbability	?	?

Copy the best solution to the clipboard

The built-in OptQuest optimizer is used to calibrate a compartment-based model of contagious disease diffusion.

Historic data, best fitting and current objective

Graph showing Current objective (blue line) and Best fitting (red line) over iterations.

Calibration - Optimization Experiment

General: Application options (will not be applied when model runs as applet)

Advanced: Maximum available memory: 128 Mb

Java machine arguments:

Command-line arguments:

Load root object from snapshot:

Imports section:

Additional class code:

Initial experiment setup: `dsInfectiousHistoric.fillFrom(InfectiousHistoric);`

Before each experiment run: `datasetCurrentObjective.reset(); datasetBestFeasibleObjective.reset();`

Before simulation run:

After simulation run:

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: ?

Only the best solution of the clipboard

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

The screenshot displays the AnyLogic Advanced software interface for calibrating an Agent Based SIR Model. The main window shows the 'Calibration of Agent Based SIR Model' with a 'Run calibration' button and a progress graph. The progress graph shows a vertical red line at approximately 0.8 on the y-axis, indicating the current iteration. The 'Current' and 'Best' columns show 'infeasible' and '?' for the iteration and objective, respectively. The 'Parameters' table shows 'ContactRate' with '?' in both columns.

The bottom console window, titled 'Calibration - Optimization Experiment', contains two tables for constraints:

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

The screenshot displays the AnyLogic Advanced [EDUCATIONAL USE ONLY] interface. The main window is titled "Calibration of Agent Based SIR Model" and features a "Run calibration" button. A table compares "Current" and "Best" values for "Iteration" (both marked as "infeasible" and "?") and "Objective" (both "?"). A "Calibration progress" graph shows a vertical red line at approximately 0.8 on the x-axis. The bottom console window, titled "Calibration - Optimization Experiment", contains two tables: "Constraints on simulation parameters" and "Requirements (are tested after a simulation run to determine whether the solution is feasible)".

Calibration - Optimization Experiment

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. The 'Calibration: Main' folder is expanded, showing various components like 'Functions', 'Analysis Data', and 'Presentation'. The 'dsInfectiousBest' component is selected.
- Calibration Experiment 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 this, there are several data points represented by colored circles and labels: 'InfectiousHistoric', 'dsInfectiousHistoric', 'dsInfectiousCurrent', 'difference', and 'dsInfectiousBest'.
- Properties Panel (Bottom Left):** Shows the 'Replications' section with a checkbox labeled 'Use replications' which is currently unchecked.
- Console (Bottom Center):** Displays the title 'Calibration - Optimization Experiment'.
- Right Panel (Right):** Contains a 'Current' section with 'Iteration: ?' and 'Objective: ↓ ?'. Below this is a 'Parameters' section with 'ContactRate ?' and 'InfectionProbability ?'. A yellow highlight is visible over the 'Parameters' section. At the bottom of this panel, there is a text box: "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 Recove".
- Palette (Far Right):** Shows a list of 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 data series. 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 of the "Calibration - Optimization Experiment" configuration. The settings are as follows:

- 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

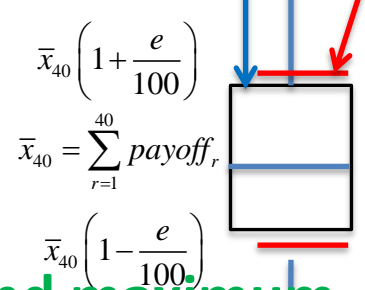
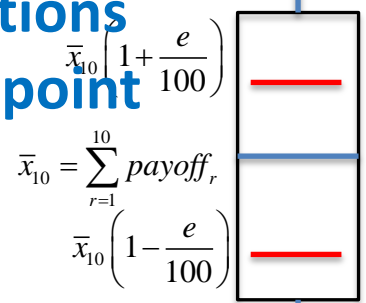
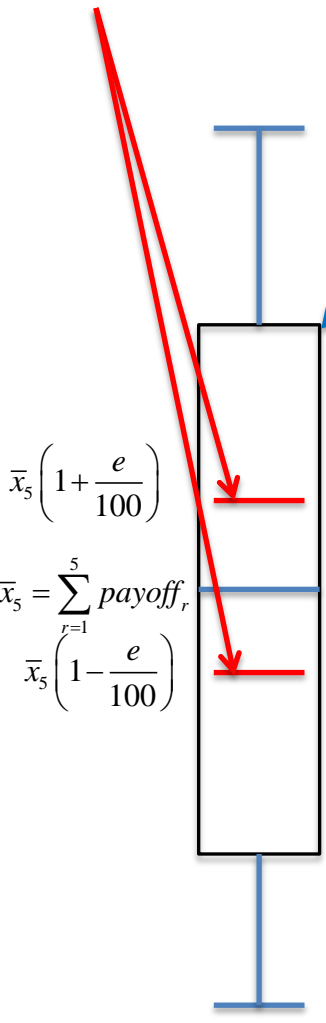
The "Confidence level" and "Error percent" fields are circled in red. A red arrow points from the text overlay to the "Confidence level" field.

Example

Bars showing that delineating values within errorPercent% of mean

Terminates because confidence interval falls within errorPercent% bars

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



After 5 replications

After 10 replications

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 panes:

- Project Explorer (Left):** Shows a hierarchical tree of model components, including variables like InfectionProbability (0.8), TotalPopulation (10000), and various datasets and presentation elements.
- Main Canvas (Center):** A grid-based workspace containing several data objects and a text box. The text box states: "These data correspond to ContactRate = 1.5, InfectionProbability = 0.4".
- Properties Panel (Bottom Left):** Shows the configuration for the "Calibration - Optimization Experiment".
 - General:** Use replications
 - Advanced:** Fixed number of replications; Varying number of replications (Stop replications after minimum replications, when confidence level is reached)
 - Replications:** Minimum replications: 10; Maximum replications: 100; Confidence level: 80%; Error percent: 0.5
- Console (Bottom Center):** Displays the title "Calibration - Optimization Experiment".
- Right Panel:** Contains a "Current" section with "Iteration: ?" and "Objective: ?" (marked as infeasible), and a "Parameters" section with "ContactRate" and "InfectionProbability" (both marked as ?).
- 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, and Environment.

Enabling Random Variation Between Realizations (“Replications”)

The screenshot displays the AnyLogic Advanced software interface, specifically the Calibration - Optimization Experiment configuration 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 points to the "InfectiousHistoric" data point.

The left sidebar shows the Project tree with various model components like InfectionProbability, TotalPopulation, and dsInfectious.

The bottom panel, titled "Calibration - Optimization Experiment", contains the following settings:

- 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) with Seed Value: 1
- Objective:** minimize; maximize; Objective function: difference()
- Optimization stop conditions:** Iteration count: 500; Automatic Stop
- Parameters:** (empty list)

The right sidebar shows the Palette with various model components like 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, and Environment.

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 (Center):** 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" properties of the selected element. It includes fields for "Iteration:" (marked as "infeasible"), "Objective:" (with a downward arrow), and "Parameters:" (ContactRate, InfectionProbability). A "Copy the best solution to the clipboard" button is also present.
- Console (Bottom):** Displays the execution log for the "Calibration - Optimization Experiment". It shows code snippets for "Before Each Experiment Run", "Before Simulation Run", "After Simulation Run", and "After Iteration Code".
- Palette (Far Right):** A vertical toolbar containing various modeling elements 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, and Environment.

The Console window shows the following code snippets:

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

Before Each Experiment Run:

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

Before Simulation Run:

After Simulation Run:

```
dsInfectiousCurrent.fillFrom( root.dsInfectious );
traceln("For this particular simulation, the difference is\t" + difference());
```

After Iteration Code:

```
if( getCurrentIteration() == getBestIteration() )
    dsInfectiousBest.fillFrom( dsInfectiousCurrent );
```

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

Calibration of Agent Based SIR Model

Run calibration

Iteration: 1

Objective: ↓ 3,343.5

Parameters

ContactRate 1.75

InfectionProbability 0.45

Copy the best solution to the clipboard

Calibration progress

Historic data, best fitting and current simulation output

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

Calibration progress

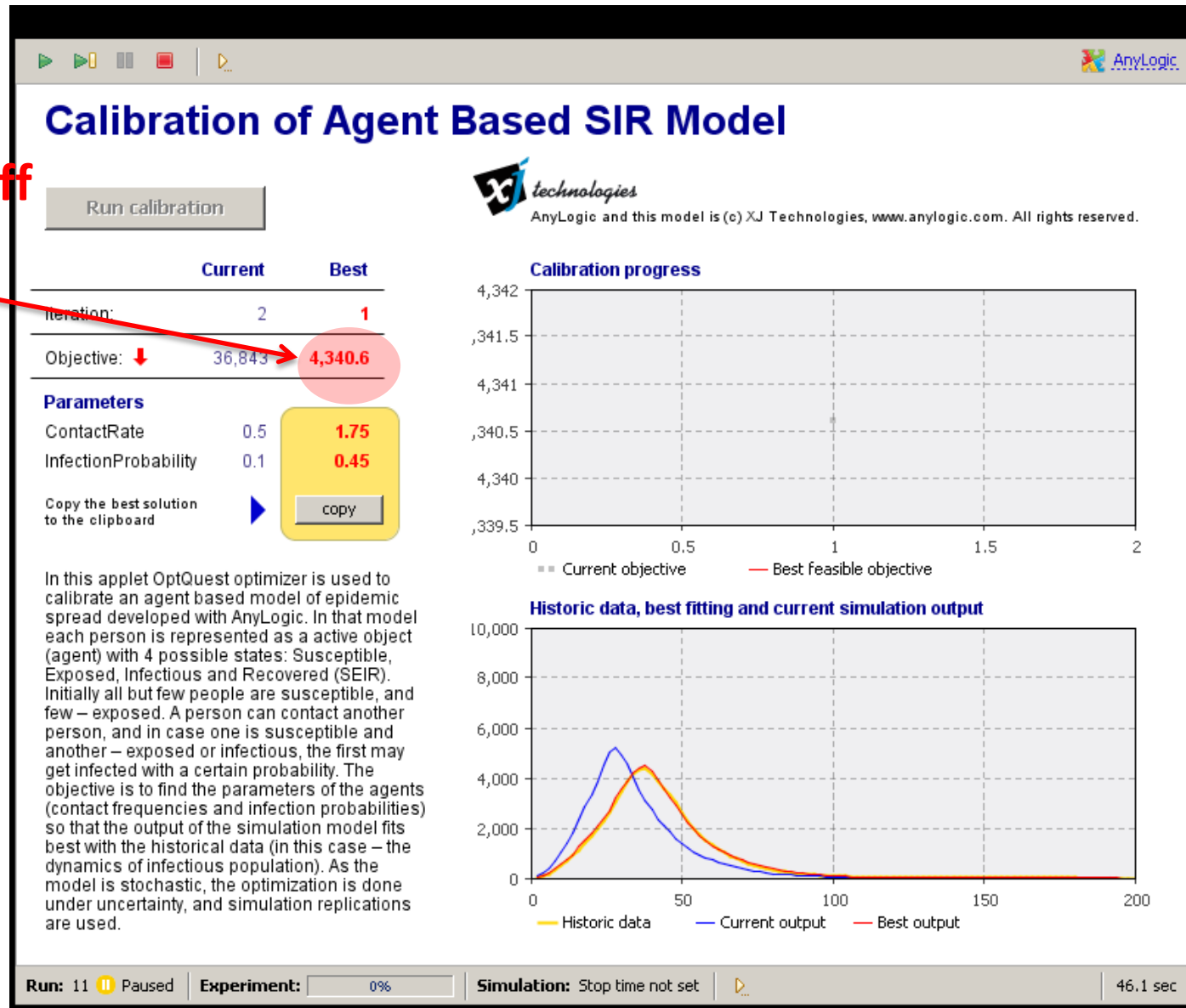
Legend: Current objective (grey square), Best feasible objective (red line)

Historic data, best fitting and current simulation output

Legend: Historic data (yellow line), Current output (blue line), Best output (red line)

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 shows the AnyLogic Advanced interface. The main window displays a list of variables and data sets, including 'datasetBestFeasibleObjective', 'dsInfectiousHistoric', 'dsInfectiousCurrent', 'difference', and 'dsInfectiousBest'. A text box in the center of the main window states: 'These data correspond to ContactRate = 1.5, InfectionProbability = 0.4'. The 'Current' tab on the right shows 'Iteration: infeasible' and 'Objective:'. The 'Parameters' section shows 'ContactRate' and 'InfectionProbability'. The 'Console' window at the bottom displays 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 average value 36843. The text below the console window reads: 'The reported payoff for the iteration is the average of the payoffs for each replication within the replication'.

Selection Cursor: X=-105, Y=376

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