Designing Simulation Experiments

W. David Kelton
Dept. of Quantitative Analysis & Operations Management
University of Cincinnati
Cincinnati, Ohio 45221-0130, USA
david.kelton@uc.edu
http://www.econqa.cba.uc.edu/~kelton
1-513-556-5499

Introduction..................................................................................................................2

2^k Factorial Designs ........................................................................................................5

Coping with Many Factors ...............................................................................................12

2^{k-p} Fractional Factorial Designs ...............................................................................13
Factor-Screening Strategies ............................................................................................16

Response Surfaces and Metamodels ...............................................................................17

Sensitivity and Gradient Estimation ................................................................................22

Optimum Seeking............................................................................................................23

Optimum-Seeking Methods ............................................................................................24
Optimum-Seeking Packages Interfaced with Simulation Software ..................................25
Introduction

Point of most simulation projects: running the model and trying to understand the results

Need to plan ahead how this will be done, to avoid time-consuming inefficiencies

A simulation study is an experiment that needs to be designed

Identify purpose of the project
   Just one system configuration? Still have issues like run length, number of runs, how to assign random numbers driving the simulation, interpretation of results
   Several “given” system configurations? Have same questions, as well as how to compare, select, or rank the alternatives
   How do changes in inputs affect outputs?
   Search for an optimal system configuration?

Identify relevant output performance measures
   Time frame of model
      Terminating (or short-run or finite horizon or transient)
      Steady-state (or long-run or infinite horizon)
   Types of output performance measures (can affect model structure)
      Means
      Variances, standard deviations
      Probabilities
      Quantiles

Use and allocation of underlying random numbers (for stochastic simulations)
   Random-number generators are controllable (fixed sequence, streams, seeds)
   Completely independent (separate random numbers) vs. re-use for variance reduction
Broader goal of simulation projects: To learn how the inputs affect the outputs

Analogy to traditional physical experiments (laboratory, industrial, agricultural):

<table>
<thead>
<tr>
<th>Example</th>
<th>Inputs (factors)</th>
<th>Outputs (responses)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chemical reaction</td>
<td>Pressure</td>
<td>Yield</td>
</tr>
<tr>
<td></td>
<td>Temperature</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Catalyst concentration</td>
<td></td>
</tr>
<tr>
<td>Growing tomatoes</td>
<td>Fertilizer</td>
<td>Yield</td>
</tr>
<tr>
<td></td>
<td>Soil pH</td>
<td>Hardiness</td>
</tr>
<tr>
<td></td>
<td>Seed hybrid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Water</td>
<td></td>
</tr>
<tr>
<td>Simulation of a manufacturing system</td>
<td>Job dispatch rule</td>
<td>Throughput</td>
</tr>
<tr>
<td></td>
<td>Number of machines</td>
<td>Time in system</td>
</tr>
<tr>
<td></td>
<td>Machines’ reliability</td>
<td>Utilizations</td>
</tr>
<tr>
<td></td>
<td>Mean downtimes</td>
<td>Queue sizes</td>
</tr>
</tbody>
</table>

Kinds of factors (inputs):
- Quantitative vs. Qualitative
- Controllable vs. Uncontrollable (in modeling, *everything* is controllable)

Simulation output performance measures are the *responses*

Goals:
- Design simulation studies to learn about the model efficiently—don’t squander runs
- Find out quickly which factors are unimportant and forget about them
- Build simple proxy models of complicated, expensive simulation models
- Use simulation to optimize system performance
- Use simulation to measure the effect of changes in factor levels
Usually there are many possible factors, responses:

<table>
<thead>
<tr>
<th>System</th>
<th>Possible Factors</th>
<th>Quantitative?</th>
<th>Qualitative?</th>
<th>Controllable?</th>
<th>Uncontrollable?</th>
<th>Possible Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supermarket checkout facility</td>
<td>Mean interarrival time</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Delay in queue</td>
</tr>
<tr>
<td></td>
<td>Mean service time</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>Time in system</td>
</tr>
<tr>
<td></td>
<td>Number of physical lanes</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Queue lengths</td>
</tr>
<tr>
<td></td>
<td>Presence of express lanes</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Checker utilizations</td>
</tr>
<tr>
<td></td>
<td>Checker adding/removing policy</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Add/remove frequency</td>
</tr>
<tr>
<td>Manufacturing line</td>
<td>Number of machines</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Parts throughput</td>
</tr>
<tr>
<td></td>
<td>Queue discipline</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Time in system</td>
</tr>
<tr>
<td></td>
<td>Buffer sizes</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Machine utilizations</td>
</tr>
<tr>
<td></td>
<td>Conveyor speeds</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Profitability</td>
</tr>
<tr>
<td></td>
<td>Machine groupings into cells</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communications network</td>
<td>Message arrival rates</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Message delays</td>
</tr>
<tr>
<td></td>
<td>Message durations</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Message throughput</td>
</tr>
<tr>
<td></td>
<td>Number of nodes</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Node, link utilizations</td>
</tr>
<tr>
<td></td>
<td>Number of links</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>System reliability</td>
</tr>
<tr>
<td></td>
<td>Protocol used</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Maintenance policy</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inventory system</td>
<td>Mean interdemand time</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>Holding cost</td>
</tr>
<tr>
<td></td>
<td>Number of items demanded</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>Shortage cost</td>
</tr>
<tr>
<td></td>
<td>Lead time from supplier</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Ordering cost</td>
</tr>
<tr>
<td></td>
<td>Reorder point</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td>Profitability</td>
</tr>
<tr>
<td></td>
<td>Reorder amount</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>Backlogged/lost orders</td>
</tr>
<tr>
<td></td>
<td>Inventory evaluation frequency</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Backlog vs. lost sales</td>
<td>✓</td>
<td>?</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Setting factor *levels* (values they can take on): No real prescription

  Qualitative—may be clear from context

  Quantitative—may set at “reasonable” levels, but that might push the boundaries

Special opportunities in simulation-based experiment:

  Everything is controllable

  Control source of randomness, exploit for variance reduction

  No need to randomize assignment of treatments to experimental units
\(2^k\) Factorial Designs

Large literature on experimental design, most applicable to simulation

Example of a design that is feasible in many simulations: \(2^k\) factorial

Have \(k\) factors (inputs), each at just two levels

Number of possible combinations of factors is thus \(2^k\)

Case of single factor \((k = 1)\):

Vary the factor (maybe at more than two levels), make plots, etc.

In general, assume \(k \geq 2\) factors—want to know about:

Effect on response(s) of each factor

Possible interactions between factors—effect of one factor depends on the level of some of the other factors

One-at-a-time strategy:

for each factor \(j = 1, 2, \ldots, k\) {
  for all \(2^{k-1}\) combinations of the other \((\neq j)\) factors {
    simulate for a low and high value of factor \(j\) (2 runs)
    compute difference in the 2 responses
  } /* \(2 \times 2^{k-1} = 2^k\) runs */
  average the \(2^{k-1}\) differences to get the average effect of factor \(j\)
} /* \(k\) times through this loop */

Total number of runs = \(k \times 2^k\)

Using \(2^k\) factorial design, can get by with \(2^k\) runs, and get more information (interactions)
**Code** each factor to a “+” and a “−” level

**Design matrix:** All possible combinations of factor levels

Example for $k = 3$ factors:

<table>
<thead>
<tr>
<th>Factor combination (Design point)</th>
<th>Factor 1</th>
<th>Factor 2</th>
<th>Factor 3</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>−</td>
<td>−</td>
<td>−</td>
<td>$R_1$</td>
</tr>
<tr>
<td>2</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>$R_2$</td>
</tr>
<tr>
<td>3</td>
<td>−</td>
<td>+</td>
<td>−</td>
<td>$R_3$</td>
</tr>
<tr>
<td>4</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>$R_4$</td>
</tr>
<tr>
<td>5</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>$R_5$</td>
</tr>
<tr>
<td>6</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>$R_6$</td>
</tr>
<tr>
<td>7</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>$R_7$</td>
</tr>
<tr>
<td>8</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>$R_8$</td>
</tr>
</tbody>
</table>

Make the 8 simulation runs, measure effects of the factors

Main effect of a factor is the average difference in the response when this factor is at its “+” level as opposed to its “−” level:

$$
e_1 = \frac{(R_2 - R_1) + (R_4 - R_3) + (R_6 - R_5) + (R_8 - R_7)}{4}
$$

Can rewrite as “Factor 1” column • “Response” column / $2^{k-1}$

Interaction effects: Does effect of a factor depend on level of others?

$1 \times 3$ interaction effect: “Factor 1” • “Factor 3” • “Response” / $2^{k-1}$

Sign of factor indicates direction of effect on response of moving that factor from its “−” to its “+” level

Statistical significance of effects estimates? (i.e., are they real?)

Luxury in simulation-based experiments:

Replicate the whole design $n$ times

Get $n$ observations on each effect

Sample mean, sample variance, confidence interval, etc., on expected effects

Effect is “real” if confidence interval misses 0

Limitations: Based on linear regression model (2-factor case)

$$E(R) = \beta_0 + \beta_{1x_1} + \beta_{2x_2} + \beta_{12x_1x_2}
$$

Interpreting main effects ($\beta_1$ and $\beta_2$) as the “change” in the response owing to a move of the corresponding factor from the “−” to the “+” level is valid only if there is no interaction ($\beta_{12} = 0$)
**Example of 2^2 Factorial Design**

Inventory model with costs for holding inventory, ordering from supplier, and shortage; response = total of holding, ordering, and shortage costs over 10 years

Demands against inventory occur at random times and are for random amounts

Review inventory at beginning of each month

If inventory level is at least \( s \), do not order from supplier
If inventory level is less than \( s \), order an amount \( d \) from supplier

Factors are \( s \) and \( d \)

Coding chart for factors:

<table>
<thead>
<tr>
<th>Factor</th>
<th>–</th>
<th>+</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>20</td>
<td>60</td>
</tr>
<tr>
<td>( d )</td>
<td>10</td>
<td>50</td>
</tr>
</tbody>
</table>

Design matrix and results:

<table>
<thead>
<tr>
<th>Factor combination (design point)</th>
<th>( s )</th>
<th>( d )</th>
<th>( s \times d )</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>–</td>
<td>–</td>
<td>+</td>
<td>141.86</td>
</tr>
<tr>
<td>2</td>
<td>+</td>
<td>–</td>
<td>–</td>
<td>141.37</td>
</tr>
<tr>
<td>3</td>
<td>–</td>
<td>+</td>
<td>–</td>
<td>112.45</td>
</tr>
<tr>
<td>4</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>146.52</td>
</tr>
</tbody>
</table>

Effects estimates:

\( e_s = 16.79 \), so moving \( s \) from 20 to 60 increases total cost by 16.79

\( e_d = -12.13 \), so moving \( d \) from 10 to 50 decreases total cost by 12.13

\( e_{sd} = 17.28 \), so having \( s \) and \( d \) at opposite levels decreases cost

If interaction is really present, then main-effects estimates’ meanings are clouded

Are the effects statistically significant? Replicated whole design \( n = 10 \) times, got \( n = 10 \) estimates of each of the effects, computed sample means, standard deviations, 90% confidence intervals:

\( e_s = 17.66 \pm 1.02 \)  All effects appear statistically significant ... practically significant?

\( e_d = -8.73 \pm 1.27 \)  Interaction seems to be present ... cannot interpret main effects directly as change in response when factor moves from its – to its + level

\( e_{sd} = 10.60 \pm 1.94 \)
Example of $2^6$ Factorial Design

Also: Machine suffers breakdowns, must undergo repair

Response: Average time in system of a part (*makespan*)

Factors and coding:

<table>
<thead>
<tr>
<th>Factor number</th>
<th>Factor description</th>
<th>– (current)</th>
<th>+ (improved)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Machining times</td>
<td>U(0.5, 0.6)</td>
<td>U(0.45, 0.54)</td>
</tr>
<tr>
<td>2</td>
<td>Inspection times</td>
<td>U(0.65, 0.70)</td>
<td>U(0.585, 0.630)</td>
</tr>
<tr>
<td>3</td>
<td>Machine uptimes</td>
<td>expo(360)</td>
<td>expo(396)</td>
</tr>
<tr>
<td>4</td>
<td>Machine repair times</td>
<td>U(8, 12)</td>
<td>U(7.2, 10.8)</td>
</tr>
<tr>
<td>5</td>
<td>Probability bad part</td>
<td>0.10</td>
<td>0.09</td>
</tr>
<tr>
<td>6</td>
<td>Queue disciplines (both)</td>
<td>FIFO</td>
<td>Shortest job first</td>
</tr>
</tbody>
</table>

Full $2^6$ factorial design involved 64 factor combinations

Replicated entire design $n = 5$ times (several hours in SIMAN on a PC)
Can see several trends:

“Two-up, two-down” pattern suggests that factor 2 (inspection times, which alternate in sign on every second run in design matrix) is important

Right half is almost a copy of left half, suggesting that factor 6 (queue discipline, which are “+” for the left half and “−” for the right half) doesn’t matter
Effects estimates (and confidence intervals around them):

Can see that factor 2 (inspection time) has big negative effect on makespan—"improving" it to "+" level would be the single most worthwhile step to take to reduce makespan

Improving factor 5 (probability of failing inspection) would have next-most-important effect on makespan

Seem not to be any significant interactions, so can directly interpret main effects
Factorial designs can be very useful, but they do have their limitations

- In order to interpret the main effects literally, must assume that expected response can be expressed as a linear function of the factors, which implies no interactions. In inventory model, the underlying regression model for the design is

\[ E[R(s, d)] = \beta_0 + \beta_s s + \beta_d d + \beta_{sd} sd \]

and least-squares estimates of the \( \beta \) coefficients are linear transformations of the effects estimates. So unless \( \beta_{sd} = 0 \), i.e., there is no interaction, the main effects of the factors alone are not the change in response given a change in the factor.

- Even if there is no interaction, the effects estimates cannot generally be interpreted as the change in response when moving the factor by a “distance” equal to the difference from its – to its + level regardless of the starting point ... unless we make the strong assumption that the response is linear everywhere, not just in the region of the factor space where we experimented.

- Results are relative to the particular values chosen for the factors, and cannot necessarily be extrapolated to other regions in the factor space.

- It’s probably not good to choose the – and + levels of a factor to be extremely far apart from each other
  - Could result in experiments for factor levels that are unrealistic in the problem context
  - Get no information on “interior” of design space between the factor levels, so we might not see nonlinearity or interactions that might be present there
Coping with Many Factors

In many studies, there are initially a lot of factors:

If \( k \) is big, then \( 2^k \) is really big

Need: Some way to reduce (screen out) the number of factors initially
   Identify the ones that seem not to have much effect (like queue discipline in above example)
   Set them at some reasonable or convenient level and forget about them

Then proceed with a more manageable (smaller) value of \( k \)

Effectively, reduces the dimensionality of the active factor space

This is useful for experimental design, as well as for metamodels and optimum seeking, discussed later
$2^{k-p}$ Fractional Factorial Designs

Full $2^k$ factorial design allows estimation of all main effects, 2-way interactions, 3-way interactions, ..., the $k$-way interaction

Give up ability to estimate interactions (especially higher order) by confounding them with each other

Confounded effects have the same estimation formula

This common formula estimates the sum of the expected confounded effects

\[ E(e_4) + E(e_{123}) \]

Thus, need to assume that the three-way interaction is not present for this estimate to be unbiased for $E(e_4)$

It often happens that three- and higher-way interactions are not as strong as main effects, so such an assumption may be reasonable, so we could regard $e_4$ as being a valid estimate of the main effect of factor 4

In return, get by with a fraction ($1/2^p$) of the runs: make only $2^{k-p}$ runs

Do just a fraction ($1/2^p$) of the $2^k$ runs for the full-factorial design

Key question: Which of the $2^k$ runs to do?

Answer: Not easy; must be careful to pick runs to get “clear” effects desired and confound uninteresting effects with each other

Fortunately, there are tables to create fractional-factorial designs via a recipe

Higher values of $p$ lead to:

- Fewer runs required (good)
- Give up ability to estimate more effects due to more confounding (bad)
Resolution of a fractional factorial design: Roman numerals III, IV, V, etc.: one way to quantify overall severity of confounding

Two effects are guaranteed not to be confounded with each other if the sum of their “ways” is $< \text{resolution of the design (“way” of a main effect is 1)}$

Resolution III:
- Main effects unconfounded with each other: $1 + 1 < 3$
- Main effects confounded with 2-way interactions: $1 + 2 \text{ not} < 3$
- 2-way interactions confounded with each other: $2 + 2 \text{ not} < 3$

Very weak designs since you cannot even trust the main-effects estimates because they can be confounded with 2-way interactions, and you cannot determine whether 2-way interactions are present since they’re confounded with each other

Resolution IV:
- Main effects unconfounded with each other: $1 + 1 < 4$
- Main effects unconfounded with 2-way interactions: $1 + 2 < 4$
- 2-way interactions confounded with each other: $2 + 2 \text{ not} < 4$

Better than resolution III, since main effects are clear of confounding with 2-way interactions, but since the 2-way interactions are confounded with each other you can’t tell if they’re present so don’t know if you can interpret the main effects (see earlier discussion on linearity of response)

Resolution V:
- Main effects unconfounded with each other: $1 + 1 < 5$
- Main effects unconfounded with 2-way interactions: $1 + 2 < 5$
- 2-way interactions unconfounded with each other: $2 + 2 < 5$

Much better than resolution IV, since you now can tell if you have 2-way interactions (they’re unconfounded with each other), so can know how to interpret the main effects

In simulation, 2-way interactions can often be present, so resolution V or higher is recommended
Constructing fractional factorial designs of desired resolution:

For number of factors \( k \), look up in a design table what resolution designs are possible (not all resolutions are possible for all values of \( k \))

Read off the value of \( p \) for the desired resolution from the table

Write out the design matrix for a full \( 2^{k-p} \) factorial design, for columns (factors) \( 1, 2, \ldots, k - p \)

Read off the definition of the other \( p \) factors (factors \( k - p + 1, k - p + 2, \ldots, k \)) and fill in these remaining columns (factors) of the design matrix accordingly

Run the factor configurations called for and compute the available effects, heeding the design’s resolution

Compute available effects as in full-factorial design

Example: For \( k = 8 \) factors there is a resolution V design with \( p = 2 \) (i.e., a 1/4 fractional design) defined by \( 7 = 1234 \) and \( 8 = 1256 \) (i.e., multiply the corresponding columns’ + and – signs together to get the columns for factors 7 and 8)

Enhanced inventory example, with four rather than two factors

Additional factors: evaluation interval (\( m = 1 \) or 2), express ordering (\( p = \) no or yes)

Replicated designs 10 times; 90% c.i.’s on expected main effects:

<table>
<thead>
<tr>
<th>Factor</th>
<th>Full ( 2^4 ) factorial design</th>
<th>( 2^{4-2} ) half fraction design</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>(-0.61 \pm 0.86)</td>
<td>(-2.34 \pm 0.90)</td>
</tr>
<tr>
<td>( d )</td>
<td>(-10.62 \pm 0.91)</td>
<td>(-10.66 \pm 0.90)</td>
</tr>
<tr>
<td>( m )</td>
<td>(1.87 \pm 0.75)</td>
<td>(1.96 \pm 0.81)</td>
</tr>
<tr>
<td>( p )</td>
<td>(41.86 \pm 0.95)</td>
<td>(33.39 \pm 0.42)</td>
</tr>
</tbody>
</table>

The full factorial displayed some significant 2-way interactions, so interpretation of main effects cannot be made literally

Machine/repair example:

Did \( 2^{6-2} \) fractional factorial design (1/4 the work, ran in 3 hours rather than 12)

Came to identical conclusions as from the full 12-hour \( 2^6 \) design

In full design, there were not significant interactions present
Factor-Screening Strategies

Specialized designs intended specifically for screening when number of factors is too large for even a highly fractionated design

Plackett-Burman: Look at effects of $k$ factors in as few as $k + 1$ design points

Supersaturated designs: Number of factors > number of design points possible to investigate
  - Balanced designs: Each column has exactly half “+” signs, half “–” signs
  - Random-balance: Sprinkle the half “+” and half “–” signs randomly within a column
  - Systematic (non-random) balanced supersaturated designs

Group-screening: Group factors together in some way dependent on model
  - Use intuition so that “+” and “–” levels of individual factors within a group are expected to affect response in the same direction
  - Move all the factors in a group up and down together
  - Calculate effects of each group
  - Screen out unimportant groups
  - Disaggregate groups

Frequency-domain methods
  - Oscillate input factors periodically during a single run, at different frequencies
  - Look at oscillation in output, match with frequencies of input oscillations to identify important factors
Response Surfaces and Metamodes

Large-scale simulations can sometimes themselves become too expensive to run

Rough proxy: A simple algebraic approximation to how a performance measure depends on the input factors

(Algebraic) model of the (simulation) model — a *metamodel*

Back to the “machine” view of a simulation:

Algebraically, output = \( f(\text{inputs}) \), for some very complicated function \( f \) that is expressed only by the simulation model and code

Idea:

- Run the actual simulation for some limited number of combinations of inputs
- Fit some kind of regression model to describe how the observed outputs depend on the input
Example of Fitting a Metamodel

Inventory simulation, costs for holding, ordering, and shortages
Response (performance measure): Average total cost per month
Factors (input decision variables): Reorder point \( s \) and order amount \( d \)
Made \( n = 10 \) replications for \( s = 0, 5, 10, ..., 100 \) and \( d = 5, 10, 15, ..., 100 \) (4,200 runs in all)
Response-surface plot:

Optimal (low-cost) \( (s, d) \) appears to be around \( (25, 35) \)
Fit a full-quadratic regression model to the simulation using 5 replications at 16 points (80 runs in all), got the fitted model

\[ 130.63 - 0.26s - 0.53d + 0.004sd + 0.009s^2 + 0.005d^2 \]
Fit same full-quadratic model, but now using 5 replications at 36 points (180 runs in all):

\[188.51 - 1.49s - 1.24d + 0.014sd + 0.007s^2 + 0.010d^2\]
Some Comments on Metamodelling

Factorial designs discussed earlier are particular forms of regression metamodells with the factors (independent variables) suitably transformed linearly.

More specific and sophisticated designs have been developed to support metamodelling.

If there are many factors, it might be good to screen out unimportant ones before metamodelling, to reduce the number of independent variables.

Most simulation models have multiple responses, so it might be possible to build separate metamodells for the different responses.

Can use variance-reduction techniques to improve a metamodel’s stability.
Sensitivity and Gradient Estimation

A frequent goal in simulation: How does the performance measure change with changes in the input parameter(s)?

Using “output = \( f \) (inputs)” model of the simulation, this is a question about the derivatives of \( f \)

Example where exact results are available:

\( M/M/1 \) queue in steady state

Arrival rate \( \lambda \), service rate \( \omega \)

Expected steady-state delay in queue is \( d(\lambda, \omega) = \frac{\lambda}{\omega^2 - \lambda \omega} \)

Partial derivatives:

\[ \frac{\partial d}{\partial \lambda} = \frac{1}{(\omega - \lambda)^2}, \quad \frac{\partial d}{\partial \omega} = -\frac{\lambda(2\omega - \lambda)}{\omega^2 (\omega - \lambda)^2} \]

Give instantaneous change in \( d \) owing to small changes in \( \lambda \) or \( \omega \)

How to do this when simulation is necessary?

Direct approach (finite differences):

- Make a run with the factor of interest in one spot
- Move it a little bit and make another run
- Compute slope (rise/run)

Indirect approach:

- Use a fitted metamodel as a proxy, take partial derivatives of it

Single-run methods:

- Perturbation analysis—move the factor during the run, track new trajectory as well as trajectory if the perturbation had not been made
- Likelihood ratios (a.k.a. score functions)

Frequency-domain methods
Optimum Seeking

Perhaps the most ambitious of all goals in simulation: Find a set of input-factor values that optimize (maximize or minimize) a key output performance measure.

Focus on inputs that are controllable in reality.

In some ways similar to comparison/selection, but there the alternative configurations were given, and here we are looking for the best configurations.

Can formulate this as a mathematical programming problem: maximize or minimize an expected output performance measure, subject to constraints on the feasible input-factor combinations, but there are challenges:

Difficult if number of input factors is large: high-dimensional factor space to search.

Objective “function” is the simulation itself, not just an algebraic formula.

Stochastic simulations produce random, or “noisy” objective-function values.

Despite challenges, this is a much-sought-after goal in many simulation studies, and there are numerous examples of attempting it.
Optimum-Seeking Methods

What underlying methodology should be used to look for an optimum set of input factors?

One approach:
- Fit a simple regression metamodel
- Optimize it with simple calculus

Might be possible to do a more economical “search” without wasting time simulating in “inferior” regions of the factor space—response surface methods

Fit a low-cost, low-class linear metamodel using only a few points

Use it to define a search direction along which further simulations are run

Active area of research, with many methods proposed and evaluated
Optimum-Seeking Packages Interfaced with Simulation Software

Practical issue of how to manage a possibly extensive search through the input-factor space, and link the search to a possibly-large simulation model.

There are several optimum-seeking packages, each with its own search method built in, interfaced with simulation-modeling software:

Most operational systems are based on heuristic-search methods.

Get displays of how the objective value improves as the search progresses.

Need to specify various parameters (like tolerances) to tell the underlying search method how “aggressively” to search, how many replications to make at a point, what the stopping criterion is for the search, etc.