Chapter 11
Tuning an Algorithm Using Design of Experiments

Enda Ridge and Daniel Kudenko

Abstract This chapter is a tutorial on using a *design of experiments* approach for tuning the parameters that affect algorithm performance. A case study illustrates the application of the method and interpretation of its results.

11.1 Introduction

This chapter presents a case study of a methodology for selecting algorithm parameter values to tune algorithm performance. It efficiently takes the experimenter from the initial situation of almost no knowledge of the algorithm's behavior to the desired situation of an accurately modeled algorithm. This provides recommendations on tuning parameter settings for given problem characteristics. The methodology is based on well-established procedures from *design of experiments* (DOE) (Montgomery 2005) that have been modified for their application to algorithm tuning (Ridge 2007). The field of DOE is defined as:

> ... a systematic, rigorous approach to engineering problem-solving that applies principles and techniques at the data collection stage so as to ensure the generation of valid, defensible, and supportable engineering conclusions. In addition, all of this is carried out under the constraint of a minimal expenditure of engineering runs, time, and money. (Croarkin and Tobias 2006)

DOE is well established theoretically and well supported in terms of software tools. The traditional areas to which DOE is applied in engineering map almost directly to the common research questions that one asks in algorithm research so

Enda Ridge
Datalab, The Technology Innovation Group, Detica Ltd., London, UK
e-mail: Enda.Ridge@Detica.com

Daniel Kudenko
Department of Computer Science, The University of York, UK
e-mail: Kudenko@cs.york.ac.uk

T. Bartz-Beielstein et al. (eds.), *Experimental Methods for the Analysis of Optimization Algorithms*, DOI 10.1007/978-3-642-02538-9_11,

DOE's power and maturity can be transferred directly to algorithm research. DOE offers efficiency in terms of the amount of data that needs to be gathered. This is critical when attempting to understand immense algorithm design spaces. All DOE conclusions are based on statistical analyses and so are supported with mathematical precision. This allays any concerns regarding subjective interpretation of results.

The case study applies DOE to an *ant colony system* (ACS) (Dorigo and Stützle 2004) for the *traveling salesperson problem* (TSP) (Lawler et al. 1995). Further details on the ACS algorithm are well covered in the literature. For our purposes here, it is sufficient to understand that ACS is a heuristic (randomized) optimization algorithm with many tuning parameters and at least two problem characteristics that affect its performance. Before reporting the case study, we first discuss some preliminaries that are important for an understanding of DOE.

11.2 Research Questions Addressed with DOE

Modeling algorithm performance is a sensible way to explore the vast design space of tuning parameter settings and their relationship to problem instances and algorithm performance. A good model can be used to quickly explore algorithm performance without resorting to expensive algorithm runs. Models permit addressing the following research questions:

- **Screening.** Which tuning parameters and which problem characteristics have no significant effect on the performance of the algorithm in terms of solution quality and solution time?
- **Ranking.** What is the relative importance of the most important tuning parameters and problem characteristics?
- **Relationship between tuning, problems, and performance.** What is the relationship between tuning parameters, problem characteristics, and the responses of solution quality and solution time? A tuning study yields a mathematical equation modeling this relationship for each response.
- **Tuned parameter settings.** What is a good set of tuning parameter settings given an instance with certain characteristics? Are these settings better than what can be achieved with randomly chosen settings? Are these settings better than alternative settings from the literature?

11.3 Experiment Designs

The design that an experimenter uses will depend on many things, including the particular research question, whether experiments are in the early stages of research, and the experimental resources available. This section focuses on the advanced designs that appear in this chapter. It begins with a simpler, more common design as this provides the necessary background for understanding the subsequent designs.

11.3.1 Full and 2^k Factorial Designs

A *full factorial* design consists of a crossing of all levels of all factors. A factor (or independent variable) is a variable that an experimenter varies. The number of levels of each factor can be two or more and need not be the same for each factor. The full factorial is an extremely powerful but expensive design. A more useful type of factorial for DOE uses k factors, each at only 2 levels. The so-called 2^k factorial design provides the smallest number of runs with which k factors can be studied in a full factorial design. Factorials have some particular advantages and disadvantages (Ostle 1963). These are worth noting given the importance that factorials play in DOE experimental design. The advantages are that:

- Greater efficiency is achieved in the use of available experimental resources in comparison with what could be learned from the same number of experiment runs in a less structured context such as a *one-factor-at-a-time* analysis,
- Information is obtained about the interactions, if any, of factors because the factor levels are all crossed with one another. An interaction is where the experimental response for a given factor level cannot be understood without also specifying the level of an interacting factor.

Of course, these advantages come at a price. As the number of factors grows, the number of combinations of factor levels (treatments) in a 2^k design rapidly overwhelms the experiment resources. Consider the case of 10 continuous factors. A naïve full factorial design for these ten factors will require a prohibitive $2^{10} = 1024$ treatments. A more efficient design is required.

11.3.2 Fractional Factorial Designs

There are benefits to the expense of a full factorial design. A 2^{10} full factorial will provide data to evaluate all the effects listed in Table 11.1.

If it is assumed that higher-order interactions are insignificant, information on the main effects and lower-order interactions can be obtained by running a fraction of the complete factorial design. This assumption is based on the *sparsity of effects principle* (Wu and Hamada 2000). This states that a system or process is likely to be most influenced by some main effects and low-order interactions and less influenced by higher-order interactions.

A judiciously chosen fraction of the treatments in a full factorial will yield insights into only the lower-order effects. This is termed a *fractional factorial*. The price we pay for the fractional factorial's reduction in number of experimental treatments is that some effects are indistinguishable from one another; they are *aliased*. Additional treatments, if necessary, can disentangle these aliased effects should an alias group be statistically significant. The advantage of the fractional factorial is that it facilitates sequential experimentation. The additional treatments and associated experiment runs need only be performed if aliased effects are statistically

Effect	Number of effects estimated
Main	10
Two-factor	45
Three-factor	120
Four-factor	210
Five-factor	252
Six-factor	210
Seven-factor	120
Eight-factor	45
Nine-factor	10
Ten-factor	1

Table 11.1: Number of each effect estimated by a full factorial design of 10 factors

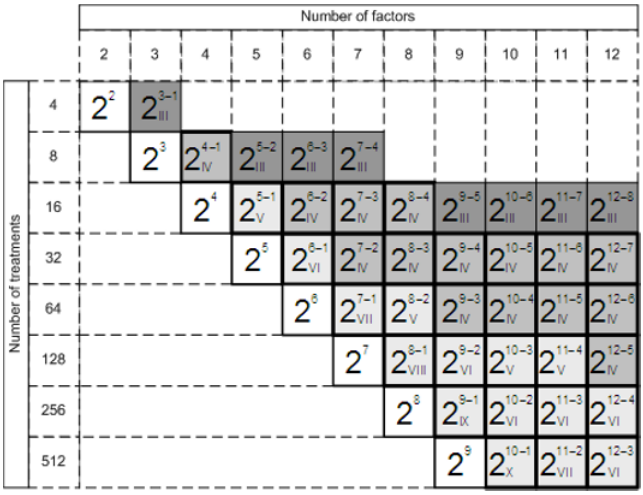
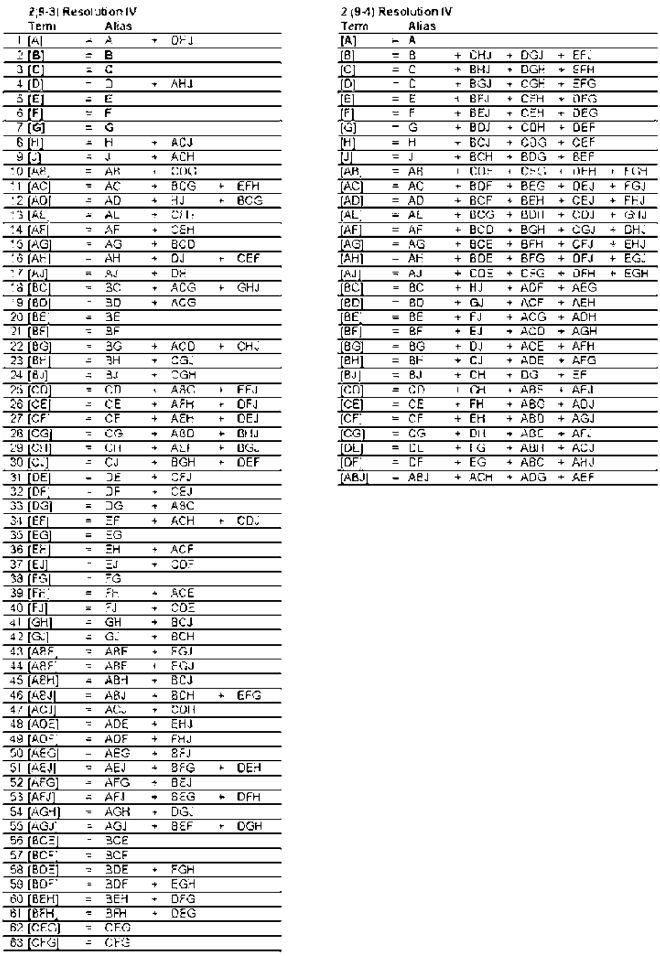


Fig. 11.1: Fractional factorial designs for 2 to 12 factors. The required number of treatments is listed on the left. Resolution III designs (do not estimate any terms) are colored darkest followed by Resolution IV designs (estimate main effects only), followed by Resolution V and higher (estimate main effects and second-order interactions)

significant. Depending on the number of factors, and consequently the design size, a range of fractional factorials can be produced from a full factorial.

The amount of higher-order effects that are aliased is described by the design's *resolution*. For Resolution III designs, all effects are aliased. Resolution IV designs have unaliased main effects but second-order effects are aliased. Resolution V designs estimate main and second-order effects without aliases. The details of how to choose a fractional factorial's treatments are beyond the scope of this chapter. It is an established algorithmic procedure that is well covered in the literature (Montgomery 2005) and is provided in all modern statistical analysis software. The fractional factorials used in this case study are summarized in Fig. 11.1 which shows the



2 (9-3) Resolution IV

Term		Alias						
1 [A]	=	A	+	DFJ				
2 [B]	=	B						
3 [C]	=	C						
4 [D]	=	D	+	AHJ				
5 [E]	=	E						
6 [F]	=	F						
7 [G]	=	G						
8 [H]	=	H	+	ADJ				
9 [J]	=	J	+	ADH				
10 [AB]	=	AB	+	CDJ				
11 [AC]	=	AC	+	BCG	+	EFH		
12 [AD]	=	AD	+	HJ	+	BCG		
13 [AE]	=	AE	+	CFJ				
14 [AF]	=	AF	+	CEH				
15 [AG]	=	AG	+	BCD				
16 [AH]	=	AH	+	DJ	+	CEF		
17 [AJ]	=	AJ	+	DH				
18 [BC]	=	BC	+	ACG	+	GHJ		
19 [BD]	=	BD	+	ACG				
20 [BE]	=	BE						
21 [BF]	=	BF						
22 [BG]	=	BG	+	ACD	+	CHJ		
23 [BH]	=	BH	+	CGJ				
24 [BJ]	=	BJ	+	CGH				
25 [CD]	=	CD	+	ABC	+	EFJ		
26 [CE]	=	CE	+	AFH	+	DFJ		
27 [CF]	=	CF	+	AEH	+	DEJ		
28 [CG]	=	CG	+	ABD	+	BHJ		
29 [CH]	=	CH	+	AJ	+	DGJ		
30 [CJ]	=	CJ	+	BGH	+	DEF		
31 [DE]	=	DE	+	CFJ				
32 [DF]	=	DF	+	CEJ				
33 [DG]	=	DG	+	ABC				
34 [EF]	=	EF	+	ACH	+	CDJ		
35 [EG]	=	EG						
36 [EH]	=	EH	+	ACF				
37 [EJ]	=	EJ	+	CDF				
38 [FG]	=	FG						
39 [FH]	=	FH	+	ACE				
40 [FJ]	=	FJ	+	CDE				
41 [GH]	=	GH	+	BCJ				
42 [GJ]	=	GJ	+	BCH				
43 [ABF]	=	ABF	+	FGJ				
44 [ABF]	=	ABF	+	FGJ				
45 [ABH]	=	ABH	+	BCJ				
46 [ABJ]	=	ABJ	+	BCH	+	EFG		
47 [ACJ]	=	ACJ	+	CDH				
48 [ADE]	=	ADE	+	EHJ				
49 [ADF]	=	ADF	+	FHJ				
50 [AEG]	=	AEG	+	BFJ				
51 [AEJ]	=	AEJ	+	BFG	+	DEH		
52 [AFG]	=	AFG	+	BEJ				
53 [AFJ]	=	AFJ	+	BEG	+	DFH		
54 [AGH]	=	AGH	+	DGJ				
55 [AGJ]	=	AGJ	+	BEF	+	DGH		
56 [BCE]	=	BCE						
57 [BCF]	=	BCF						
58 [BDE]	=	BDE	+	FGH				
59 [BDF]	=	BDF	+	EGH				
60 [BEH]	=	BEH	+	DFG				
61 [BFH]	=	BFH	+	DEG				
62 [CFG]	=	CFG						
63 [CFG]	=	CFG						

2 (9-4) Resolution IV

Term		Alias						
[A]	=	A						
[B]	=	B	+	CHJ	+	DGJ	+	EFJ
[C]	=	C	+	BHJ	+	DGH	+	EFH
[D]	=	D	+	BGJ	+	CGH	+	EFG
[E]	=	E	+	BFJ	+	CFH	+	DFG
[F]	=	F	+	BEJ	+	CEH	+	DEG
[G]	=	G	+	BDJ	+	CDH	+	DEF
[H]	=	H	+	BCJ	+	CDG	+	CEF
[J]	=	J	+	BCH	+	BDG	+	BEF
[AB]	=	AB	+	CDJ	+	CEH	+	DEH	+	FGH
[AC]	=	AC	+	BDF	+	BEG	+	DEJ	+	FGJ
[AD]	=	AD	+	BCF	+	BEH	+	CEJ	+	FHJ
[AE]	=	AE	+	BCG	+	BDH	+	CDJ	+	GHJ
[AF]	=	AF	+	BCD	+	BGH	+	CGJ	+	DHJ
[AG]	=	AG	+	BCE	+	BFH	+	CFJ	+	EHJ
[AH]	=	AH	+	BDE	+	BFG	+	DFJ	+	EGJ
[AJ]	=	AJ	+	CDE	+	CFG	+	DFH	+	EGH
[BC]	=	BC	+	HJ	+	ADF	+	AEG
[BD]	=	BD	+	GJ	+	ACF	+	AEH
[BE]	=	BE	+	FJ	+	ACG	+	ADH
[BF]	=	BF	+	EJ	+	ACD	+	AGH
[BG]	=	BG	+	DJ	+	ACE	+	AFH
[BH]	=	BH	+	CJ	+	ADE	+	AFG
[BJ]	=	BJ	+	CH	+	DG	+	EF
[CD]	=	CD	+	GH	+	ABF	+	AFJ
[CE]	=	CE	+	FH	+	ABC	+	ADJ
[CF]	=	CF	+	EH	+	ABD	+	AGJ
[CG]	=	CG	+	DH	+	ABE	+	AFJ
[DE]	=	DE	+	FG	+	ABH	+	ACJ
[DF]	=	DF	+	EG	+	ABC	+	AHJ
[ABJ]	=	ABJ	+	ACH	+	ADG	+	AEF

Fig. 11.2: Effects and alias chains for a 2(9-3) resolution IV design and a 2(9-4) resolution IV design

relationship between number of factors, design resolution, and associated number of experiment treatments.

A resolution V design is preferable when resources allow because it tells us what second-order effects are present without the need for additional treatments and experiment runs. It is informative to consider the two available resolution IV designs for 9 factors in Fig. 11.2 as examples of the importance of examining alias structure.

The 2(9-4) design requires 32 treatments while the 2(9-3) is more expensive with 64 treatments. The cheaper 2(9-4) design has 8 of its 9 main effects aliased with 3 third-order interactions. The 2(9-3) design has only 4 of its 9 main effects aliased with a single third-order interaction. The second-order interactions are almost all

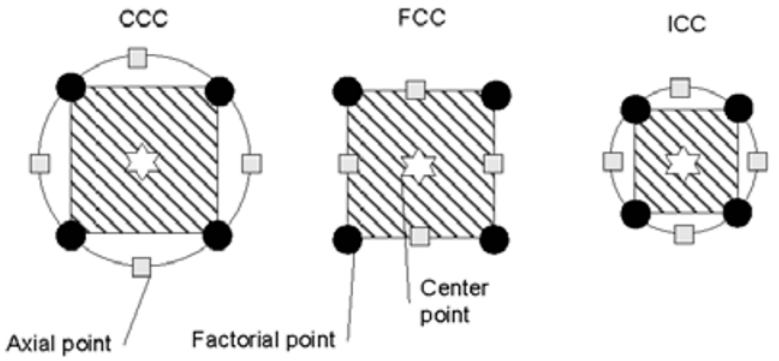


Fig. 11.3: Central composite designs for building response surface models. From left to right these designs are the circumscribed central composite (CCC), the face-centred composite (FCC) and the inscribed central composite (ICC). The design space is represented by the shaded area. The factorial points are black circles and the star points are grey squares

aliased in the more expensive 2(9-3) design but the aliasing is more favorable than in the cheaper 2(9-4) design.

11.3.3 Response Surface Designs

There are several types of experiment design for building response surface models. This chapter's case study uses *central composite designs* (CCD). A CCD contains an imbedded factorial (or fractional factorial design). This is augmented with both center points and a group of so-called *star points* (or axial points) that allow estimation of curvature in the resulting model. There are three types of central composite design, illustrated in Fig. 11.3.

The choice of design depends on the nature of the factors to study:

- **Circumscribed central composite (CCC).** In this design, the star points establish new extremes for the low and high settings for all factors. These designs require 5 levels for each factor. Augmenting an existing factorial or resolution V fractional factorial design with star points can produce this design.
- **Inscribed central composite (ICC).** For those situations in which the limits specified for factor settings are truly limits, the ICC design uses the factor settings as the star points and creates a factorial or fractional factorial design within those limits. This design also requires 5 levels of each factor.

Design	Treatments	% saving of treatments
Full	512	
2 (9-5) III	16	97
2(9-4) IV	32	94
2(9-3) IV	64	88
2(9-2) VI	128	75

Design*	Treatments	% saving of treatments
Full	531	
Half	275	50
Quarter	147	75
Min Run	65	91

* FCC with 1 centre point

Table 11.2: Savings in experiment runs. The savings for screening designs are on the left and the savings for response surface designs are on the right. In both cases, fractional factorial designs offer enormous savings in number of treatments over the full factorial alternative. "Half" and "quarter" refer to the fraction of the full design used. "Min Run" is a further extension to this concept permitting even greater run savings

- **Face-centred Composite (FCC).** In this design, the star points are at the center of each face of the factorial space. This design requires just 3 levels of each factor.

An existing factorial or resolution V design from the screening stage can be augmented with appropriate star points to produce the CCC and FCC designs. This is not the case with the ICC and so it is less useful in a sequential experimentation scenario.

11.3.4 Efficiency of Fractional Factorial Designs

Table 11.2 makes explicit the huge savings in experiment runs when using a fractional factorial design instead of a full factorial design.

11.4 Error, Significance, Power, and Replicates

Two types of error can be committed when testing hypotheses (Montgomery 2005, p. 35). If the null hypothesis is rejected when it is actually true, then a *type I error* has occurred. If the null hypothesis is not rejected when it is false then a *type II error* has occurred. These error probabilities are given special symbols:

- $\alpha = \Pr\{\text{Type I error}\} = \Pr\{\text{reject } H_0 | H_0 \text{ true}\}$
- $\beta = \Pr\{\text{Type II error}\} = \Pr\{\text{fail to reject } H_0 | H_0 \text{ false}\}$

In the context of type II errors, it is more convenient to use the *power* of a test, where

$$\text{Power} = 1 - \beta = \Pr\{\text{reject } H_0 |\ H_0 \text{ false}\}.$$

It is therefore desirable to have a test with a low α and a high power. The probability of a Type I Error is often called the *significance level* of a test. The particular significance level depends on the requirements of the experimenter and, in a research context, on the conventional acceptable level. Unfortunately, with so little adaptation of statistical methods to the analysis of heuristics, there are few guidelines on what value to choose. The power of a test is usually set to 80% by convention. The reason for this choice is due to diminishing returns. It requires an exponentially increasing number of replicates to increase power beyond about 80% and there is little advantage to the additional power this confers.

Miles[1] describes the relationship between significance level, effect size, sample size, and power using an analogy with searching.

- **Significance level:** This is the probability of thinking we have found something when it is not really there. It is a measure of how willing we are to risk a type I error.
- **Effect size:** The size of the effect in the population. The bigger it is, the easier it will be to find. This is a measure of the practical significance of a result, preventing us claiming a statistically significant result that has little consequence (Rardin and Uzsoy 2001).
- **Sample size:** A larger sample size leads to a greater ability to find what we were looking for. The harder we look, the more likely we are to find it.

The critical point regarding this relationship is that what we are looking for is always going to be there—it might just be there in such small quantities that we are not bothered about finding it. Conversely, if we look hard enough, we are guaranteed to find what we are looking for. Power analysis allows us to make sure that we have looked reasonably hard enough to find it. A typical experiment design approach is to agree the significance level and choose an effect size based on practical experience and experiment goals. Given these constraints, the sample size is increased until sufficient power is reached. If a response has a high variability then a larger sample size will be required.

Different statistical tests and different experiment designs involve different power calculations. These calculations can become quite involved and the details of their calculation are beyond the scope of this chapter. Power calculations are supplied with most good-quality statistical analysis software.

11.5 Benchmarking the Experimental Testbed

Clearly, all machines for computational experiments can differ widely. There are differences in processor speeds, memory sizes, chip types, operating systems, operating system versions, and in the case of Java different versions of different virtual

[1] "Getting the sample size right: a brief introduction to power analysis", http://www.jeremymiles.co.uk/misc/power

			Time (s)						
Instance	Size	Repetitions	116	253	111	156	188	136	96
E1k.0	1000	1000	5.45	4.38	5.25	5.00	3.31	4.81	3.37
E3k.0	3000	316	5.67	4.61	7.61	5.25	3.78	5.23	3.75
E10k.0	10000	100	6.91	7.25	8.81	6.44	4.99	6.64	5.32
E31k.0	31000	32	11.77	16.52	13.41	11.20	9.48	11.00	10.84
E100k.0	100000	10	22.86	26.53	26.77	21.03	10.87	19.85	12.82
E316k.0	316000	3	28.61	32.05	34.50	27.61	12.56	25.86	14.70
E1M.0	1000000	1	39.52	44.80	49.23	38.03	16.55	35.44	19.31

Table 11.3: Data from the DIMACS benchmarking of the experiment testbed. Running times are presented for the 7 experimental machines with IDs 116, 253, 111, 156, 188, 136, and 96

machines. Even if machines are identical in terms of all of these aspects, they may still differ in terms of running background processes such as virus checkers. This is the unfortunate reality of the majority of computational research environments. Furthermore, such differences will almost certainly occur in the computational resources of other researchers who attempt to reproduce or extend previous work of others. These differences necessitate the benchmarking of the experimental testbed.

Reproducibility of results is the motivation for benchmarking. Other researchers can reproduce the benchmarking process on their own experimental machines. They can thus better interpret the CPU times reported in this research by scaling them in relation to their own benchmarking results. This mitigates the decline in relevance of reported CPU times with inevitable improvements in technology.

The clear and simple benchmarking procedure of the DIMACS (Goldwasser et al. 2002) challenge was applied for this chapter's case study. The results are presented in Table 11.3. If other researchers reproduce the DIMACS procedure on their own machines then their numbers can be compared with Table 11.3 to scale the results.

11.6 Case Study

This section reports the chapter's case study on tuning the ACS algorithm with DOE. It is a template for how such DOE experiments could be reported since standardized reporting in other fields has greatly helped the interpretation of research results.

11.6.1 Problem Instances

All TSP instances were of the Euclidean symmetric type. In the Euclidean TSP, cities are points with integer coordinates in the two-dimensional plane. A cost matrix defines the distances between all cities in the problem instance. The TSP problem instances ranged in size from 300 cities to 500 cities with cost matrix standard deviation ranging from 10 to 70. All instances had a cost matrix mean of 100. The

same instances were used for each replicate of a design point. Instances were generated with a version of the publicly available portmgen problem generator from the DIMACS challenge (Goldwasser et al. 2002).

11.6.2 Stopping Criterion

The choice of how to halt an experiment affects the results of an algorithm and thus the conclusions that can be drawn from the experiments. In this case study, experiments were halted after a stagnation stopping criterion. Stagnation was defined as a fixed number of iterations in which no improvement in solution value had been obtained. Responses were measured at several levels of stagnation during an experiment run: 50, 100, 150, 200, and 250 iterations. This facilitated examining the data at alternative stagnation levels to ensure that conclusions were the same regardless of stagnation level. An examination of the descriptive statistics verifies that the stagnation level did not have a large effect on the response values and therefore the conclusions after a 250 iteration stagnation should be the same as after lower iteration stagnations.

11.6.3 Response Variables

For experiments with algorithms, the response variables usually reflect some measure of solution quality and some measure of solution time. Two response variables were measured. The time in seconds to the end of an experiment reflects the *solution time*. The *relative error* from a known optimum reflects the solution quality. Concorde (Applegate et al. 2003) was used to calculate the optima of the generated instances. One may wonder why one would use a heuristic on a problem where the optimal solution can be calculated. The intention here is to evaluate a design of experiments methodology in a controlled manner.

11.6.4 Factors, Levels and Ranges

11.6.4.1 Held-Constant Factors

There are several held-constant factors. Local search, a technique typically used in combination with ACS, was omitted. All instances had a cost matrix mean of 100. The computation_limit parameter (Dorigo and Stützle 2004) was fixed at being limited to the candidate list length as this resulted in significantly lower solution times.

Factor	Name	Type	Low level	High level
A	Alpha	Numeric	1	13
B	Beta	Numeric	1	13
C	antsFraction	Numeric	1.00	110.00
D	nnFraction	Numeric	2.00	20.00
E	q0	Numeric	0.01	0.99
F	Rho	Numeric	0.01	0.99
G	rhoLocal	Numeric	0.01	0.99
H	solutionConstruction	Categoric	parallel	sequential
J	antPlacement	Categoric	random	same
K	pheromoneUpdate	Categoric	bestSoFar	bestOfIteration
L	problemSize	Numeric	300	500
M	problemStDev	Numeric	10.00	70.00

Table 11.4: Design factors for the tuning case study with ACS. The factor ranges are also given

11.6.4.2 Nuisance Factors

A limitation on the available computational resources necessitated running experiments across a variety of machines with slightly different specifications. Runs were executed in a randomized order across these machines to counteract any uncontrollable nuisance factors due to the background processes and differences in machine specification.

11.6.4.3 Design Factors

The design factors are the algorithm tuning parameters and problem characteristics whose relationship to performance metrics will be modeled by the DOE response surfaces. This case study examined 12 design factors. These factors and their high and low levels are listed in Table 11.4. Note that the two problem instance characteristics are included in the experiment design as we are modeling the relationship between the tuning parameters and various problems.

These parameters are discussed in further detail in the literature (Ridge 2007, Dorigo and Stützle 2004). For this case study, we are interested in the tuning parameters primarily as experiment design factors.

11.6.4.4 Experiment Design, Significance, Power and Replicates

The experiment design was a minimum run resolution V face-centred composite with six center points. A significance (alpha) level of 5% is used in this case study.[2]

[2] The value 5% is not a universally recommended significance level. The choice of alpha will depend on the statistical confidence required to the results. As discussed earlier, the cost of increased confidence is an increased number of experiment replicates.

Iterations		Time	Relative Error
50	Mean	65.33	11.01
	StDev	194.96	19.49
	Max	3131.77	125.84
	Min	0.17	0.55
	Actual Effect Size of 0.2 stdevs	**38.99**	**3.90**
100	Mean	136.38	10.73
	StDev	469.81	19.09
	Max	7825.44	124.20
	Min	0.28	0.55
	Actual Effect Size of 0.2 stdevs	**93.96**	**3.82**
150	Mean	204.39	10.57
	StDev	681.54	18.95
	Max	12075.97	124.20
	Min	0.38	0.47
	Actual Effect Size of 0.2 stdevs	**136.31**	**3.79**
200	Mean	270.25	10.47
	StDev	906.16	18.85
	Max	15423.77	124.20
	Min	0.50	0.46
	Actual Effect Size of 0.2 stdevs	**181.23**	**3.77**
250	Mean	341.36	10.40
	StDev	1121.20	18.81
	Max	15573.66	123.74
	Min	0.61	0.46
	Actual Effect Size of 0.2 stdevs	**224.24**	**3.76**

Table 11.5: Descriptive statistics for the ACS FCC design. The actual detectable effect size of 0.2 standard deviations is shown for each response and for each stagnation point

The number of replicates in the design is increased and further data are gathered until a power of 80% is reached. The effect size detectable at this combination of significance and power is then examined. Replicates were introduced into the design until an appropriate effect size was detectable. This approach of increasing replicates is known as a *work-up* procedure (Czarn et al. 2004). Note that it is more common to fix effect size and significance, increasing replicates until sufficient power is reached. The principle nonetheless remains the same.

The size of effect that could feasibly be detected depended on the particular response and the particular experiment design. Table 11.5 gives the descriptive statistics for the collected data and the actual effect size for each response. The design could achieve sufficient power with 5 replicates while detecting an effect of size 0.2 standard deviations in the response value.

At this stage, we have sufficient data to build a response surface model of *each individual* response. This model will be able to detect effects of the given effect size with a given confidence and power. We will ultimately be combining both the response models' recommendations into a simultaneous tuning of all the responses.

11.6.5 Model Fitting

The highest-order response surface model that can be generated from the FCC design used in this case study is quadratic. All lower-order models (linear and 2-factor interaction) are generated. If the model is not significant, it is removed from consideration and the next highest order model is examined for significance.

1. **Find important effects.** A stepwise linear regression is performed on the chosen model to estimate its coefficients. The stepwise regression identifies the model terms that can safely be removed, giving the most *parsimonious* model possible.
2. **Diagnosis.** The usual diagnostics of the proposed linear regression model are performed. This ensures that the model assumptions have not been violated.
3. **Normality.** A normal plot of studentized residuals should be approximately a straight line. Deviations from this may indicate that a transformation of the response is appropriate.
4. **Constant variance.** A plot of Studentized residuals against predicted response values should be a random scatter. Patterns such as a "megaphone" may indicate the need for a transformation of the response.
5. **Time-dependent effects.** A plot of Studentised residuals against run order should be a random scatter. Any trend indicates the influence of some time-dependent nuisance factor that was not countered with randomization.
6. **Model fit.** A plot of predicted values against actual response values will identify particular treatment combinations that are not well predicted by the model. Points should align along the 45° axis.
7. **Leverage and influence.** Leverage measures the influence of an individual design point on the overall model. A plot of leverage for each treatment indicates any problem data points.
8. A plot of **Cook's distance** against treatment measures how much the regression changes if a given case is removed from the model.

If the model passes these tests then its proposed coefficients can be accepted. If the model does not pass its diagnostics, there are two main options:

1. **Response transformation.** A transformation of the response may be required. Transformation simply means taking some function of the response variable such as the log or square root. The appropriate transformation can be identified using a *Box-Cox plot*.
2. **Outliers.** If the transformed response is still failing the diagnostics, it may be that there are outliers in the data. These should be identified and removed from the data. In this case study, outliers were deleted and the model building repeated until the models passed the diagnostics. Of the total data, 122 data points ($\sim$5%) were removed when analyzing the models of *relative error* and *time*.

It is always good practice to independently confirm the models' accuracy on some real data, different from the data used to generate the model.

As in traditional DOE, *confirmation* is achieved by running experiments at new randomly chosen points in the design space and comparing the actual data with the model's predictions. Confirmation is not a new rigorous experiment and analysis in itself but rather a quick informal check. In the case of an algorithm, these randomly chosen points in the design space equate to new problem instances and new randomly chosen combinations of tuning parameters. The methodology is as follows:

1. **Treatments.** A number of treatments are chosen where a treatment consists of a new problem instance and a new set of randomly chosen tuning parameter values with which the instance will be solved.
2. **Generate instances.** The required problem instances are generated.
3. **Random run order.** A random run order is generated for the treatments and a given number of replicates. Three replicates are often enough to give an estimate of how variable the response is for a given treatment. We are conducting this confirmation to ensure that our subjective decisions in the model building were correct.
4. **Prediction intervals.** The collected data for each response is compared with their respective models' 95% high and low prediction intervals (Montgomery 2005, p. 394-396).[3] Two criteria upon which our satisfaction with the models (and thus confidence in their predictions) can be judged are (Ridge and Kudenko 2007):
    - **Conservative:** we should prefer models that provide consistently higher predictions of relative error and longer solution time than those actually observed. We typically wish to minimize these responses and so a conservative model will predict these responses to be higher than their true value.
    - **Matching trend:** we should prefer models that match the trends in heuristic performance. The model's predictions of the parameter combinations that give the best and worst performance should match the combinations that yield the actual algorithm's observed best and worst performance.
5. **Confirmation.** If the models are not a satisfactory predictor of the actual algorithm then the experimenter must return to the model-building phase and attempt to improve the model.

The randomly chosen treatments produced actual algorithm responses with the descriptives listed in Table 11.6.

The large ranges of each response reinforce the motivation for correct parameter tuning as there is clearly a high cost in incorrectly tuned parameters. Figure 11.4 illustrates the 95% prediction intervals and actual confirmation data for the response surface models of *relative error* and *time*.

Looking at the predictions in general we see that time was sometimes better predicted than was the relative error. The *solution time* model matches all the trends

[3] The model's $p\%$ prediction interval is the range in which you can expect any individual value from the actual algorithm to fall into $p\%$ of the time. The prediction interval will be larger (a wider spread) than a confidence interval about averages since there is more scatter in individual values than in averages.

Iterations		Time	Relative Error
100	Mean	70.14	7.01
	StDev	84.14	3.48
	Max	528.28	17.65
	Min	1.55	3.12
150	Mean	109.80	6.88
	StDev	130.37	3.41
	Max	774.39	17.65
	Min	2.17	2.77
200	Mean	169.66	6.75
	StDev	220.76	3.38
	Max	1084.58	16.92
	Min	2.83	2.77
250	Mean	220.19	6.69
	StDev	287.57	3.35
	Max	1652.54	16.84
	Min	3.45	2.77

Table 11.6: Descriptive statistics for the confirmation of the ACS tuning. The response data is from runs of the actual algorithm on the randomly generated confirmation treatments

in the actual data. The *relative error* model however exhibits some false peaks and misses some actual peaks.

11.6.6 Results

11.6.6.1 Screening and Relative Importance of Factors

Figures 11.5 and 11.6 give the ranked ANOVAs of the *relative error* and *time* models from the analysis. The terms have been rearranged in order of decreasing sum of squares so that the largest contributor to the models comes first.

Looking first at the *relative error* rankings of Fig. 11.5, we see that the least important main effects are L-antPlacement, A-alpha, F-rho, and M-pheromoneUpdate.

By far the most important terms are the main effects of the exploration/exploitation tuning parameter (E) and the candidate list length tuning parameter (D) as well as their interaction. This is a very important result because it shows that candidate list length, a parameter that we have often seen set at a fixed value or not used, is actually one of the most important parameters to set correctly.

Looking at the *time* rankings of Fig. 11.6, we see that L-antPlacement was completely removed from the model. The least important main effects were then F-rho and A-alpha.

By far the most important tuning parameters are the number of ants and the lengths of their candidate lists. This is quite intuitive as the number of processing is directly related to these parameters. The result regarding the cost of the amount of ants is particularly important because the number of ants does not have a relatively

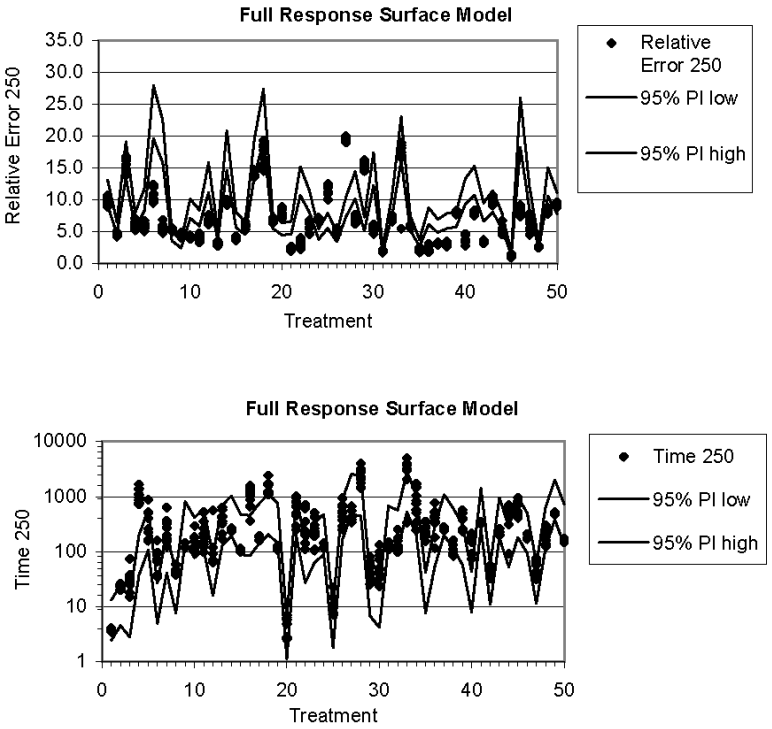


Fig. 11.4: The 95% prediction intervals for the ACS response surface model of relative error and time. The horizontal axis is the randomly generated treatment. The vertical axis is the relative error or time response

strong effect on solution quality. The extra time cost of using more ants will not result in gains in solution quality. This is an important result because it methodically confirms the often recommended parameter setting of setting the number of ants equal to a small number (usually 10).

11.6.6.2 Tuning

Since the response surface models are mathematical functions of the tuning parameters, it is possible to numerically optimize the models' responses by varying the tuning parameters. This allows us to produce the most efficient process. There are several possible optimization goals. We may wish to achieve a response with a given value (target value, maximum or minimum). Alternatively, we may wish that the response always falls within a given range (*relative error* less than 10%). More usually, we may wish to optimize several responses because of the algorithm compromise of quality and time. In the literature, tuning rarely deals with both so-

Rank	Term	Sum of squares	F value	p value
1	J-problemStDev	182.23	121362.13	< 0.0001
2	E-q0	64.87	43205.00	< 0.0001
3	D-nnFraction	22.01	14656.74	< 0.0001
4	DE	17.71	11794.89	< 0.0001
5	BD	8.86	5901.77	< 0.0001
6	EJ	4.61	3068.80	< 0.0001
7	B-beta	4.46	2967.32	< 0.0001
8	AD	4.36	2904.26	< 0.0001
9	BE	3.84	2554.84	< 0.0001
10	H-problemSize	3.66	2439.84	< 0.0001
11	AJ	3.38	2250.48	< 0.0001
12	AB	3.26	2174.28	< 0.0001
13	CE	2.76	1836.57	< 0.0001
14	G-rhoLocal	2.57	1714.47	< 0.0001
15	D^2	2.45	1631.98	< 0.0001
16	DJ	2.31	1540.95	< 0.0001
17	AF	2.26	1504.79	< 0.0001
18	E^2	2.24	1492.39	< 0.0001
19	BJ	2.21	1468.81	< 0.0001
20	B^2	2.13	1417.47	< 0.0001
21	C-antsFraction	1.93	1286.26	< 0.0001
22	EG	1.88	1253.72	< 0.0001
23	CD	1.69	1125.00	< 0.0001
24	CJ	1.50	1001.98	< 0.0001
25	EF	1.30	862.88	< 0.0001
26	EH	1.23	820.42	< 0.0001
27	DG	0.98	649.57	< 0.0001
28	K-solutionConstruction	0.84	562.08	< 0.0001
29	AG	0.42	279.16	< 0.0001
30	CF	0.38	254.96	< 0.0001
31	H^2	0.32	215.21	< 0.0001
32	M-pheromoneUpdate	0.28	188.89	< 0.0001
33	G^2	0.26	172.66	< 0.0001
34	A^2	0.24	162.14	< 0.0001
35	FH	0.21	141.53	< 0.0001
36	EM	0.21	137.62	< 0.0001
37	F-rho	0.20	134.68	< 0.0001
38	BH	0.20	132.31	< 0.0001
39	DF	0.20	131.48	< 0.0001
40	GH	0.19	124.68	< 0.0001
41	F^2	0.18	122.13	< 0.0001
42	J^2	0.15	100.38	< 0.0001
43	AH	0.14	94.24	< 0.0001
44	A-alpha	0.12	80.01	< 0.0001
45	FJ	0.12	77.53	< 0.0001
46	EK	0.10	63.99	< 0.0001
47	HJ	0.09	58.21	< 0.0001
48	DK	0.06	42.16	< 0.0001
49	JK	0.06	42.03	< 0.0001
50	DH	0.06	41.72	< 0.0001
51	AE	0.06	37.78	< 0.0001
52	FM	0.05	33.60	< 0.0001
53	HK	0.05	32.45	< 0.0001
54	CG	0.04	28.05	< 0.0001
55	BM	0.04	25.03	< 0.0001
56	DM	0.04	25.02	< 0.0001
57	GK	0.04	24.44	< 0.0001
58	JM	0.03	22.86	< 0.0001
59	BC	0.03	22.76	< 0.0001
60	GJ	0.03	19.37	< 0.0001
61	CM	0.03	17.58	< 0.0001
62	BG	0.03	17.42	< 0.0001
63	CH	0.02	16.61	< 0.0001
64	FG	0.02	9.99	0.0016
65	KM	0.01	7.98	0.0048
66	GL	0.01	7.51	0.0062
67	BK	0.01	7.49	0.0062
68	GM	0.01	6.05	0.0140
69	EL	0.01	5.58	0.0183
70	L-antPlacement	0.01	4.39	0.0363
71	FL	0.00	2.92	0.0878
72	BF	0.00	2.75	0.0972

Fig. 11.5: Relative error–time ANOVA of relative error response from the full model. The table lists the remaining terms in the model after stepwise regression in order of decreasing sum of squares

Rank	Term	Sum of squares	F value	p value
1	C-antsFraction	1214.31	35326.80	< 0.0001
2	C^2	248.18	7219.98	< 0.0001
3	H-problemSize	122.02	3549.80	< 0.0001
4	D-nnFraction	31.19	907.44	< 0.0001
5	E-q0	4.88	141.91	< 0.0001
6	DH	4.57	132.87	< 0.0001
7	EM	3.37	98.14	< 0.0001
8	HJ	2.75	80.00	< 0.0001
9	K-solutionConstructior	2.59	75.23	< 0.0001
10	M-pheromoneUpdate	2.44	71.03	< 0.0001
11	EF	1.95	56.76	< 0.0001
12	BD	1.44	41.91	< 0.0001
13	DE	1.43	41.68	< 0.0001
14	GM	1.31	38.21	< 0.0001
15	DM	1.23	35.79	< 0.0001
16	CD	1.23	35.73	< 0.0001
17	EG	1.14	33.10	< 0.0001
18	CG	1.04	30.19	< 0.0001
19	BM	0.63	18.18	< 0.0001
20	CH	0.59	17.20	< 0.0001
21	CF	0.50	14.58	0.0001
22	JM	0.45	13.21	0.0003
23	FJ	0.43	12.41	0.0004
24	AE	0.42	12.36	0.0004
25	G-rhoLocal	0.35	10.30	0.0013
26	AB	0.33	9.88	0.0019
27	B-beta	0.31	9.09	0.0026
28	AG	0.30	8.64	0.0033
29	HK	0.27	7.79	0.0053
30	BE	0.25	7.34	0.0068
31	BC	0.24	7.08	0.0078
32	FM	0.24	6.88	0.0088
33	DK	0.22	6.29	0.0122
34	FH	0.19	5.39	0.0203
35	AH	0.18	5.31	0.0213
36	AJ	0.18	5.20	0.0227
37	GH	0.17	4.91	0.0268
38	BJ	0.17	4.86	0.0276
39	AC	0.15	4.29	0.0385
40	EJ	0.14	4.19	0.0408
41	J-problemStD	0.13	3.89	0.0547
42	BG	0.11	3.26	0.0712
43	AF	0.11	3.06	0.0804
44	HM	0.09	2.76	0.0966
45	A-alpha	0.01	0.28	0.5963
46	F-rho	0.01	0.22	0.6386

Fig. 11.6: Relative error–time ANOVA of time response from the full model. The table lists the remaining terms in the model after stepwise regression in order of decreasing sum of squares

lution quality and solution time simultaneously and so neglects this compromise. A technique from DOE allows multiple response models to be simultaneously tuned.

The desirability function approach (Montgomery 2005, Croarkin and Tobias 2006) is a widely used industrial method for optimizing multiple responses. The basic idea is that a process with many quality characteristics is completely unacceptable if any of those characteristics are outside some desired limits. For each response Y_i, a *desirability function* $d_i(Y_i)$ assigns a number between 0 and 1 to the possible values of the response Y_i; $d_i(Y_i) = 0$ is a completely undesirable value, and $d_i(Y_i) = 1$ is an ideal response value. These individual k desirabilities are combined into an overall desirability D using a geometric mean:

$$D = (d_1(Y_1) \times d_2(Y_2) \times \ldots \times d_k(Y_k))^{1/k} . \tag{11.1}$$

A particular class of desirability function was proposed by Derringer and Suich (1980). Let L_i and U_i be the lower and upper limits, respectively, of response i. Let T_i be the target value. If the target value is a maximum then

$$d_i = \begin{cases} 0 & y_i < L_i \\ \left(\frac{y_i - L_i}{T_i - L_i}\right)^r & L_i \leq y_i \leq T_i \\ 1 & y_i > T_i \end{cases} \tag{11.2}$$

If the target is a minimum value then

$$d_i = \begin{cases} 1 & y_i < T_i \\ \left(\frac{U_i - y_i}{U_i - T_i}\right)^r & L_i \leq y_i \leq T_i \\ 0 & y_i > U_i \end{cases} \tag{11.3}$$

The value r adjusts the shape of the desirability function. A value of $r = 1$ is linear. A value of $r > 1$ increases the emphasis of being close to the target value. A value of $0 < r < 1$ decreases this emphasis.

The multiple responses of *solution time* and *relative error* are expressed in terms of *desirability functions*. The *overall desirability* is then the geometric mean of the individual desirabilities. A numerical optimization is applied to the response surface models' equations such that the desirability is maximized. Typically, we specify the optimization in algorithm research with the dual goals of minimizing both solution error and solution time, while allowing all algorithm-related factors to vary within their design ranges.[4] Equal priority is given to the dual goals in this tutorial but these priorities can be varied. Recall that problem characteristics are also factors in the model since we want to establish the relationship between these problem charac-

[4] It is important to note that optimization of desirability does not necessarily lead to parameter recommendations that yield optimal algorithm performance. Desirability functions are a geometric mean of the desirability of each individual response. Furthermore, a response surface model is an interpolation of the responses from various points in the design space. There is therefore no guarantee that the recommended parameters result in optimal performance; they only result in tuned performance that is better than performance in most of the design space.

teristics, the algorithm parameters, and the performance responses. It does not make sense to include these problem characteristic factors in the optimization. The optimization process would naturally select the easiest problems as part of its solution. We therefore needed to choose fixed combinations of the problem characteristics and perform the numerical optimizations for each of these combinations. A sensible choice of such combinations is a three-level factorial of the characteristics, although any level factorial is possible depending on available resources. A more detailed description of these methods follows:

1. **Combinations of problem characteristics.** A three-level factorial combination of the problem characteristics is created. In the case of two characteristics, this creates 9 combinations of problem characteristics.
2. **Numerical optimization.** For each of these problem characteristic combinations, the Nelder-Mead simplex algorithm (Nelder and Mead 1965) was used for numerical optimization of overall desirability. The optimization goal is to minimize *both* the solution error response and the solution runtime response. The problem characteristics are fixed at the values corresponding to the 3 level factorial combinations.
3. **Choose the best solution.** When the optimization has completed, the solution of the parameter settings with the highest desirability is kept and the others are discarded. Note that there may be several solutions of very similar desirability but with differing factor settings. This is due to the nature of the multiobjective optimization and the possibility of many regions of interest.
4. **Further refine the solution.** Engineering judgement and experience may lead us to further refine the most desirable solution. For example, in this tutorial we will round off integer-valued parameters that are exponents to the nearest integer value. This is because exponents are expensive to compute and our pilot studies showed little gain in solution quality for this price.

This optimization procedure has recommended parameter settings for 9 locations covering the problem space defined in Table 11.7. Of course, a user requiring more refined parameter recommendations will have to run this optimization procedure for the problem characteristics of the scenario to hand. Optimization of desirability is done on the response surface equations. Other expensive algorithm runs beside those of the design points do not have to be executed.

The rankings of the ANOVA terms has already highlighted the factors that have little effect on the responses. For example, beta is always low, except when the problem standard deviation is high. The exploration/exploitation threshold q0 is always at a maximum of 0.99, implying that exploitation is always preferred to exploration. AntsFraction is always low. The remaining unimportant factors take on a variety of values in the model desirability optimization.

Table 11.7: Full relative error–time model results of desirability optimization. The table lists the recommended parameter values for combinations of problem size and problem standard deviation. The expected time and relative error are listed with the desirability value

Size	StDev	alpha	beta	antsFraction	nnFraction	q0	rho	rhoLocal	solutionConstruction	antPlacement	pheromoneUpdate	Time05	Relative Error	Desirability
300	10	8	2	1.00	1.00	0.99	0.69	0.96	parallel	random	bestSoFar	1.15	0.46	0.96
300	40	13	5	1.00	1.00	0.98	0.95	0.28	sequential	random	bestSoFar	1.46	1.24	0.86
300	70	1	11	1.00	20.00	0.98	0.05	0.70	parallel	random	bestSoFar	1.77	2.18	0.80
400	10	8	4	1.00	1.00	0.99	0.11	0.81	parallel	random	bestSoFar	2.42	0.46	0.92
400	40	13	6	2.19	1.16	0.97	0.99	0.03	parallel	random	bestOfItera	2.83	1.33	0.82
400	70	1	11	1.61	20.00	0.98	0.01	0.07	parallel	random	bestOfItera	4.92	2.59	0.73
500	10	7	3	1.13	1.00	0.99	0.86	0.01	parallel	same	bestOfItera	4.88	0.38	0.88
500	40	13	7	1.00	1.00	0.99	0.99	0.48	parallel	random	bestSoFar	4.25	1.35	0.80
500	70	1	10	1.04	19.78	0.99	0.05	0.01	parallel	same	bestOfItera	9.24	2.54	0.70

11.6.7 Discussion

The following conclusions are drawn from the ACS tuning study:

- **Unimportant factors: Ant placement not important.** The type of ant placement has no significant effect on ACS performance in terms of solution quality or solution time. **Alpha not important.** Alpha has no significant effect on ACS performance in terms of solution quality or solution time. This confirms the common recommendation in the literature of setting alpha equal to 1. **Rho not important.** Rho has no significant effect on ACS performance in terms of solution quality or solution time. This is a new result for ACS. **Pheromone Update Ant not important.** The ant used for pheromone updates is ranked highly for solution time.
- **Most important tuning parameters.** The most important ACS tuning parameters are the heuristic exponent B-beta, the number of ants C-antsFraction, the length of candidate lists D-nnFraction, the exploration/exploitation threshold E-q0, and rhoLocal.
- **Minimum order model.** A model that is of at least quadratic order is required to model ACS solution quality and ACS solution time. This is a new result for ACS and shows that a one-factor-at-a-time approach is not an appropriate way to tune the performance of ACS.
- **Relationship between tuning, problems, and performance.** The model of *relative error–time* was a good predictor of ACS performance across the entire design space.

11.6.8 Summary

The strengths of this chapter's methodologies come from the strengths of DOE. The methodologies are adapted from well established and tested methodologies used in other fields and are therefore proven on decades of scientific and industrial experience. The fractional factorial experiment designs provide a vast saving in experiment runs. Because DOE and response surface models build a model of performance across the whole design space, many research questions can be explored. Numerical optimization of this surface can quickly recommend tuning parameter settings for different weightings of the responses of interest. One may obtain settings appropriate for long runtimes and high quality or short runtimes and lower levels of solution quality. All of these questions are answered on the same model without the need to rerun experiments.

DOE is not a panacea for the myriad difficulties that arise in the empirical analysis of algorithms. Despite the efficiency of the DOE designs, running sufficient experiments to gather sufficient data is still computationally expensive. Of course, the experiments would have been orders of magnitude more expensive had a less sophisticated approach been used.

It is hoped that this chapter has convinced the reader of the merits of the DOE approach. The researcher who embraces these methodologies will have at their disposal an established, efficient, rigorous, reproducible approach for making strong conclusions about the relationship between algorithm tuning parameters, problem characteristics, and performance.

References

Applegate D, Bixby R, Chvatal V, Cook W (2003) Implementing the Dantzig-Fulkerson-Johnson algorithm for large traveling salesman problems. Mathematical Programming Series B 97(1-2):91–153

Croarkin C, Tobias P (eds) (2006) NIST/SEMATECH e-Handbook of Statistical Methods. National Institute of Standards and Technology, URL `http://www.itl.nist.gov/div898/handbook/`

Czarn A, MacNish C, Vijayan K, Turlach B, Gupta R (2004) Statistical Exploratory Analysis of Genetic Algorithms. IEEE Transactions on Evolutionary Computation 8(4):405–421

Derringer G, Suich R (1980) Simultaneous Optimization of Several Response Variables. Journal of Quality Technology 12(4):214–219

Dorigo M, Stützle T (2004) Ant Colony Optimization. The MIT Press, MA

Goldwasser M, Johnson DS, McGeoch CC (eds) (2002) Proceedings of the Fifth and Sixth DIMACS Implementation Challenges. American Mathematical Society

Lawler EL, Lenstra JK, Kan AHGR, Shmoys DB (eds) (1995) The Traveling Salesman Problem - A Guided Tour of Combinatorial Optimization. Wiley Series in Discrete Mathematics and Optimization, NY

Montgomery DC (2005) Design and Analysis of Experiments, 6th edn. Wiley
Nelder J, Mead R (1965) A simplex method for function minimization. The Computer Journal 7
Ostle B (1963) Statistics in Research, 2nd edn. Iowa State University Press
Rardin RL, Uzsoy R (2001) Experimental Evaluation of Heuristic Optimization Algorithms: A Tutorial. Journal of Heuristics 7(3):261–304
Ridge E (2007) Design of Experiments for the Tuning of Optimisation Algorithms. Phd thesis, Department of Computer Science, The University of York
Ridge E, Kudenko D (2007) Analyzing Heuristic Performance with Response Surface Models: Prediction, Optimization and Robustness. In: Proceedings of the Genetic and Evolutionary Computation Conference, ACM, pp 150–157
Wu J, Hamada M (2000) Experiments: Planning, analysis, and parameter design optimization. Wiley, NY