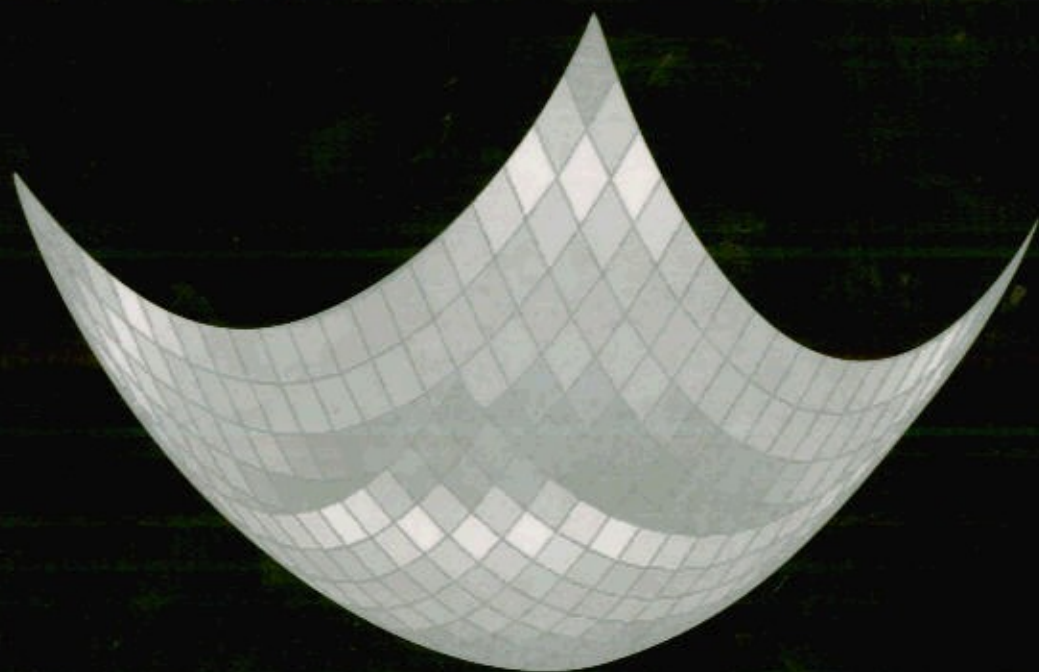


Edited by  
Bo Einarsson

---

# Accuracy and Reliability in Scientific Computing



siam

# Contents

<b>List of Contributors</b>	<b>xiii</b>
<b>List of Figures</b>	<b>xv</b>
<b>List of Tables</b>	<b>xix</b>
<b>Preface</b>	<b>xxi</b>
<b>I PITFALLS IN NUMERICAL COMPUTATION</b>	<b>1</b>
<b>1 What Can Go Wrong in Scientific Computing?</b>	<b>3</b>
<i>Bo Einarsson</i>	
1.1 Introduction . . . . .	3
1.2 Basic Problems in Numerical Computation . . . . .	3
1.2.1 Rounding . . . . .	4
1.2.2 Cancellation . . . . .	4
1.2.3 Recursion . . . . .	5
1.2.4 Integer overflow . . . . .	6
1.3 Floating-point Arithmetic . . . . .	6
1.3.1 Initial work on an arithmetic standard . . . . .	6
1.3.2 IEEE floating-point representation . . . . .	7
1.3.3 Future standards . . . . .	9
1.4 What Really Went Wrong in Applied Scientific Computing! . . . . .	9
1.4.1 Floating-point precision . . . . .	9
1.4.2 Illegal conversion between data types . . . . .	11
1.4.3 Illegal data . . . . .	11
1.4.4 Inaccurate finite element analysis . . . . .	11
1.4.5 Incomplete analysis . . . . .	12
1.5 Conclusion . . . . .	12
<b>2 Assessment of Accuracy and Reliability</b>	<b>13</b>
<i>Ronald F. Boisvert, Ronald Cools, Bo Einarsson</i>	
2.1 Models of Scientific Computing . . . . .	13
2.2 Verification and Validation . . . . .	15

2.3	Errors in Software . . . . .	17
2.4	Precision, Accuracy, and Reliability . . . . .	20
2.5	Numerical Pitfalls Leading to Anomalous Program Behavior . . . . .	22
2.6	Methods of Verification and Validation . . . . .	23
2.6.1	Code verification . . . . .	24
2.6.2	Sources of test problems for numerical software . . . . .	26
2.6.3	Solution verification . . . . .	28
2.6.4	Validation . . . . .	31
<b>3</b>	<b>Approximating Integrals, Estimating Errors, and Giving the Wrong Solution for a Deceptively Easy Problem</b>	<b>33</b>
	<i>Ronald Cools</i>	
3.1	Introduction . . . . .	33
3.2	The Given Problem . . . . .	34
3.3	The First Correct Answer . . . . .	34
3.4	View Behind the Curtain . . . . .	36
3.5	A More Convenient Solution . . . . .	36
3.6	Estimating Errors: Phase 1 . . . . .	38
3.7	Estimating Errors: Phase 2 . . . . .	40
3.8	The More Convenient Solution Revisited . . . . .	40
3.9	Epilogue . . . . .	41
<b>4</b>	<b>An Introduction to the Quality of Computed Solutions</b>	<b>43</b>
	<i>Sven Hammarling</i>	
4.1	Introduction . . . . .	43
4.2	Floating-point Numbers and IEEE Arithmetic . . . . .	44
4.3	Why Worry About Computed Solutions? . . . . .	46
4.4	Condition, Stability, and Error Analysis . . . . .	50
4.4.1	Condition . . . . .	50
4.4.2	Stability . . . . .	56
4.4.3	Error analysis . . . . .	60
4.5	Floating-point Error Analysis . . . . .	64
4.6	Posing the Mathematical Problem . . . . .	70
4.7	Error Bounds and Software . . . . .	71
4.8	Other Approaches . . . . .	74
4.9	Summary . . . . .	75
<b>5</b>	<b>Qualitative Computing</b>	<b>77</b>
	<i>Françoise Chaitin-Chatelin, Elisabeth Travesas-Cassan</i>	
5.1	Introduction . . . . .	77
5.2	Numbers as Building Blocks for Computation . . . . .	77
5.2.1	Thinking the unthinkable . . . . .	78
5.2.2	Breaking the rule . . . . .	78
5.2.3	Hypercomputation inductively defined by multiplication . . . . .	78
5.2.4	The Newcomb–Borel paradox . . . . .	79
5.2.5	Effective calculability . . . . .	80

5.3	Exact Versus Inexact Computing . . . . .	81
5.3.1	What is calculation? . . . . .	81
5.3.2	Exact and inexact computing . . . . .	82
5.3.3	Computer arithmetic . . . . .	82
5.3.4	Singularities in exact and inexact computing . . . . .	83
5.3.5	Homotopic deviation . . . . .	83
5.3.6	The map $\varphi : z \rightarrow \rho(F_z)$ . . . . .	86
5.3.7	Graphical illustration . . . . .	87
5.4	Numerical Software . . . . .	88
5.4.1	Local error analysis in finite precision computations . . . . .	88
5.4.2	Homotopic deviation versus normwise perturbation . . . . .	89
5.4.3	Application to Krylov-type methods . . . . .	90
5.5	The Lévy Law of Large Numbers for Computation . . . . .	91
5.6	Summary . . . . .	92

**II DIAGNOSTIC TOOLS 93**

**6 PRECISE and the Quality of Reliable Numerical Software 95**

*Françoise Chaitin-Chatelin, Elisabeth Travesias-Cassan*

6.1	Introduction . . . . .	95
6.2	Reliability of Algorithms . . . . .	96
6.3	Backward Error Analysis . . . . .	96
6.3.1	Consequence of limited accuracy of data . . . . .	98
6.3.2	Quality of reliable software . . . . .	98
6.4	Finite Precision Computations at a Glance . . . . .	99
6.5	What Is PRECISE? . . . . .	99
6.5.1	Perturbation values . . . . .	101
6.5.2	Perturbation types . . . . .	102
6.5.3	Data metrics . . . . .	103
6.5.4	Data to be perturbed . . . . .	103
6.5.5	Choosing a perturbation model . . . . .	103
6.6	Implementation Issues . . . . .	105
6.7	Industrial Use of PRECISE . . . . .	107
6.8	PRECISE in Academic Research . . . . .	108
6.9	Conclusion . . . . .	108

**7 Tools for the Verification of Approximate Solutions to Differential Equations 109**

*Wayne H. Enright*

7.1	Introduction . . . . .	109
7.1.1	Motivation and overview . . . . .	109
7.2	Characteristics of a PSE . . . . .	110
7.3	Verification Tools for Use with an ODE Solver . . . . .	111
7.4	Two Examples of Use of These Tools . . . . .	112
7.5	Discussion and Future Extensions . . . . .	119

<b>III</b>	<b>TECHNOLOGY FOR IMPROVING ACCURACY AND RELIABILITY</b>	<b>123</b>
<b>8</b>	<b>General Methods for Implementing Reliable and Correct Software</b>	<b>125</b>
	<i>Bo Einarsson</i>	
8.1	Ada . . . . .	127
	<i>Brian Wichmann, Kenneth W. Dritz</i>	
8.1.1	Introduction and language features . . . . .	127
8.1.2	The libraries . . . . .	128
8.1.3	The Numerics Annex . . . . .	130
8.1.4	Other issues . . . . .	132
8.1.5	Conclusions . . . . .	135
8.2	C . . . . .	136
	<i>Craig C. Douglas, Hans Petter Langtangen</i>	
8.2.1	Introduction . . . . .	136
8.2.2	Language features . . . . .	136
8.2.3	Standardized preprocessor, error handling, and debugging	138
8.2.4	Numerical oddities and math libraries . . . . .	139
8.2.5	Calling Fortran libraries . . . . .	140
8.2.6	Array layouts . . . . .	140
8.2.7	Dynamic data and pointers . . . . .	141
8.2.8	Data structures . . . . .	141
8.2.9	Performance issues . . . . .	142
8.3	C++ . . . . .	142
	<i>Craig C. Douglas, Hans Petter Langtangen</i>	
8.3.1	Introduction . . . . .	142
8.3.2	Basic language features . . . . .	143
8.3.3	Special features . . . . .	145
8.3.4	Error handling and debugging . . . . .	146
8.3.5	Math libraries . . . . .	147
8.3.6	Array layouts . . . . .	147
8.3.7	Dynamic data . . . . .	147
8.3.8	User-defined data structures . . . . .	147
8.3.9	Programming styles . . . . .	148
8.3.10	Performance issues . . . . .	148
8.4	Fortran . . . . .	149
	<i>Van Snyder</i>	
8.4.1	Introduction . . . . .	149
8.4.2	History of Fortran . . . . .	149
8.4.3	Major features of Fortran 95 . . . . .	149
8.4.4	Features of Fortran 2003 . . . . .	151
8.4.5	Beyond Fortran 2003 . . . . .	153
8.4.6	Conclusion . . . . .	153
8.5	Java . . . . .	153
	<i>Ronald F. Boisvert, Roldan Pozo</i>	
8.5.1	Introduction . . . . .	153
8.5.2	Language features . . . . .	154

8.5.3	Portability in the Java environment . . . . .	155
8.5.4	Performance challenges . . . . .	156
8.5.5	Performance results . . . . .	158
8.5.6	Other difficulties encountered in scientific programming in Java . . . . .	160
8.5.7	Summary . . . . .	162
8.6	Python . . . . .	162
<b><i>Craig C. Douglas, Hans Petter Langtangen</i></b>		
8.6.1	Introduction . . . . .	162
8.6.2	Basic language features . . . . .	163
8.6.3	Special features . . . . .	166
8.6.4	Error handling and debugging . . . . .	167
8.6.5	Math libraries . . . . .	167
8.6.6	Array layouts . . . . .	168
8.6.7	Dynamic data . . . . .	169
8.6.8	User defined data structures . . . . .	169
8.6.9	Programming styles . . . . .	170
8.6.10	Performance issues . . . . .	170
<b>9</b>	<b>The Use and Implementation of Interval Data Types</b>	<b>173</b>
<b><i>G. William Walster</i></b>		
9.1	Introduction . . . . .	173
9.2	Intervals and Interval Arithmetic . . . . .	173
9.2.1	Intervals . . . . .	174
9.2.2	Interval arithmetic . . . . .	174
9.3	Interval Arithmetic Utility . . . . .	175
9.3.1	Fallible measures . . . . .	175
9.3.2	Enter interval arithmetic . . . . .	176
9.4	The Path to Intrinsic Compiler Support . . . . .	177
9.4.1	Interval-specific operators and intrinsic functions . . . . .	179
9.4.2	Quality of implementation opportunities . . . . .	184
9.5	Fortran Code Example . . . . .	191
9.6	Fortran Standard Implications . . . . .	193
9.6.1	The interval-specific alternative . . . . .	193
9.6.2	The enriched module alternative . . . . .	194
9.7	Conclusions . . . . .	194
<b>10</b>	<b>Computer-assisted Proofs and Self-validating Methods</b>	<b>195</b>
<b><i>Siegfried M. Rump</i></b>		
10.1	Introduction . . . . .	195
10.2	Proofs and Computers . . . . .	195
10.3	Arithmetical Issues . . . . .	200
10.4	Computer Algebra Versus Self-validating Methods . . . . .	203
10.5	Interval Arithmetic . . . . .	204
10.6	Directed Roundings . . . . .	206
10.7	A Common Misconception About Interval Arithmetic . . . . .	209

10.8	Self-validating Methods and INTLAB . . . . .	215
10.9	Implementation of Interval Arithmetic . . . . .	220
10.10	Performance and Accuracy . . . . .	228
10.11	Uncertain Parameters . . . . .	230
10.12	Conclusion . . . . .	239
<b>11</b>	<b>Hardware-assisted Algorithms</b> . . . . .	<b>241</b>
	<i>Craig C. Douglas, Hans Petter Langtangen</i>	
11.1	Introduction . . . . .	241
11.2	A Teaser . . . . .	242
11.3	Processor and Memory Subsystem Organization . . . . .	243
11.4	Cache Conflicts and Trashing . . . . .	245
11.5	Prefetching . . . . .	245
11.6	Pipelining and Loop Unrolling . . . . .	246
11.7	Padding and Data Reorganization . . . . .	247
11.8	Loop Fusion . . . . .	248
11.9	Bitwise Compatibility . . . . .	249
11.10	Useful Tools . . . . .	250
<b>12</b>	<b>Issues in Accurate and Reliable Use of Parallel Computing in Numerical Programs</b> . . . . .	<b>253</b>
	<i>William D. Gropp</i>	
12.1	Introduction . . . . .	253
12.2	Special Features of Parallel Computers . . . . .	253
	12.2.1 The major programming models . . . . .	254
	12.2.2 Overview . . . . .	254
12.3	Impact on the Choice of Algorithm . . . . .	255
	12.3.1 Consequences of latency . . . . .	255
	12.3.2 Consequences of blocking . . . . .	257
12.4	Implementation Issues . . . . .	258
	12.4.1 Races . . . . .	258
	12.4.2 Out-of-order execution . . . . .	259
	12.4.3 Message buffering . . . . .	260
	12.4.4 Nonblocking and asynchronous operations . . . . .	261
	12.4.5 Hardware errors . . . . .	262
	12.4.6 Heterogeneous parallel systems . . . . .	262
12.5	Conclusions and Recommendations . . . . .	262
<b>13</b>	<b>Software-reliability Engineering of Numerical Systems</b> . . . . .	<b>265</b>
	<i>Mladen A. Vouk</i>	
13.1	Introduction . . . . .	265
13.2	About SRE . . . . .	266
13.3	Basic Terms . . . . .	267
13.4	Metrics and Models . . . . .	269
	13.4.1 Reliability . . . . .	269
	13.4.2 Availability . . . . .	274

---

13.5	General Practice . . . . .	277
13.5.1	Verification and validation . . . . .	277
13.5.2	Operational profile . . . . .	279
13.5.3	Testing . . . . .	280
13.5.4	Software process control . . . . .	284
13.6	Numerical Software . . . . .	284
13.6.1	Acceptance testing . . . . .	285
13.6.2	External consistency checking . . . . .	286
13.6.3	Automatic verification of numerical precision (error propagation control) . . . . .	287
13.6.4	Redundancy-based techniques . . . . .	289
13.7	Basic Fault-tolerance Techniques . . . . .	294
13.7.1	Check-pointing and exception handling . . . . .	294
13.7.2	Recovery through redundancy . . . . .	295
13.7.3	Advanced techniques . . . . .	297
13.7.4	Reliability and performance . . . . .	298
13.8	Summary . . . . .	298
	<b>Bibliography</b>	<b>301</b>
	<b>Index</b>	<b>335</b>