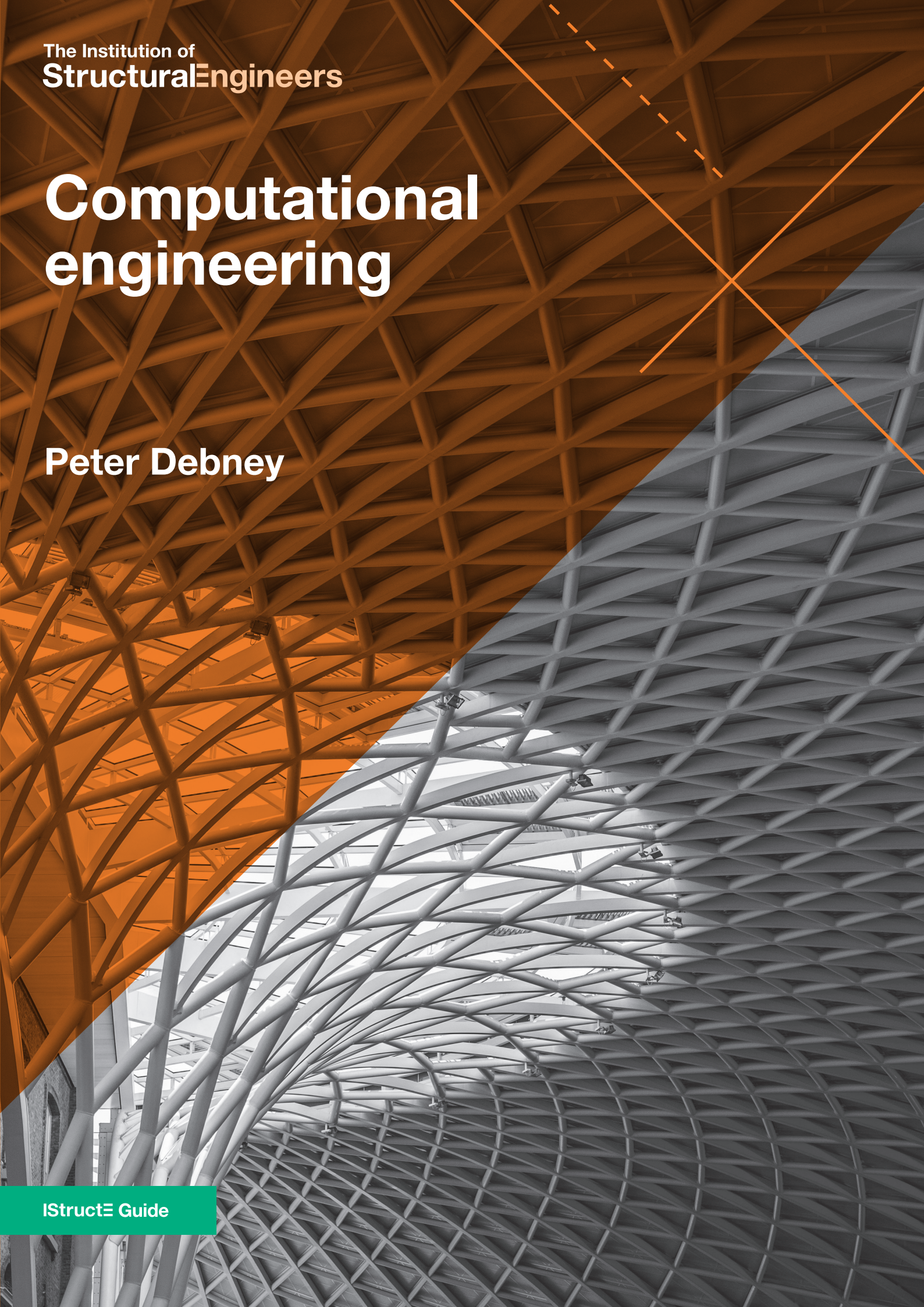
The Institution of Structural Engineers

Computational engineering

Peter Debney



IStructE Guide

The Institution of Structural Engineers

Computational engineering

Author

P Debney BEng(Hons) DipComp(Open) CEng FIStructE (Arup)

Reviewers

D Brohn CEng FIStructE (New Paradigm Solutions Ltd)
G Evans BSc(Hons) PhD CEng FICE FIStructE MBCS (Constructex) *Technical Products Panel*
R Feigin BEng(Hons) BSc MIStructE CEng MIEAust (AECOM)
C Hickey MMath(Hons) (Arup)
P Jeffries MEng(Hons) CEng MIED (Ramboll)
R Kannan PhD (Arup)
J Leach MEng CEng FIStructE MICE (AECOM)
T Q Li PhD MIstructE CEng (Arup)
I A MacLeod BSc PhD FIStructE FICE (Prof. Em., University of Strathclyde)
S Melville BEng MSc DIC CEng MIStructE FRSA (Format Engineers)
P Shepherd MA(Cantab) PhD PGCAPP CEng CMath CSci MICE FIMA SFHEA (University of Bath)
W Wild MEng(Hons) MIStructE CEng (Arup)

Publishing

L Baldwin BA(Hons) DipPub (The Institution of Structural Engineers)

Published by The Institution of Structural Engineers
International HQ, 47–58 Bastwick Street, London EC1V 3PS, United Kingdom
T: +44(0)20 7235 4535
E: mail@istructe.org
W: www.istructe.org

First published (version 1.0) October 2020

978-1-906335-44-1 (print)
978-1-906335-45-8 (pdf)

Contents

Foreword

An alien looking down at the occupants of planet Earth during the COVID-19 pandemic might have marvelled at the adaptability and resourcefulness of humankind… changing in the space of a few days from the traditional, physical face-to-face relationships of our hunter-gather origins to a new world heavily dependent on virtual remote-working, sharing information, friendships, culture, humour, shopping and news online, with no real physical equivalent. I fear for my fast-disappearing legs. This foreword is written and transmitted, from my house, in the midst of a lockdown, in a way that would have been impossible until the development of sophisticated and publicly available computational tools. I've checked the facts, remotely; I've read the original source material; looked at conflicting arguments; checked a couple of formulae and run some numerical variations; and I'm about to have a meeting with my design team all without leaving my seat. This has only become possible in the last decade or two of our planet's 13.5 billion year history, and probably there's no going back: we are at a watershed and it signals an end to the "old way of working".

Four decades ago, when I began as a practising engineer, I was told that "the mark of a good engineer was to be able to produce 10 pages of neat, accurate calculations per day". Slowly, the first finite element programmes churned away, with the DEC 10, a machine the size of a small building churning out reams of 16 digit numbers that you would hunt through to try and spot the one critical condition… No previews, and a mistake took 24 hours to re-run. Even in 2005 the idea of being able to shift an entire workforce out of its expensive offices to work at home without major disruption would have been fanciful… yet our office has made the transition seamlessly over a weekend. And many have said they like commuting virtually, and have no plans to go back to physical travel.

Such is the power and convenience, and the adaptability of our digital tools. Eric Schmidt, former CEO of Google, once remarked we are now in a partnership between humans and machines in which humans should continue to do what they do best. On Artificial Intelligence, Jason Collins, a behavioural and data scientist from PwC Australia, inverts it more pithily in a down-under-ish fashion: "Before humans become the standard way in which we make decisions," he writes, "we need to consider the risks and ensure implementation of human decision-making systems does not cause widespread harm."

Of course we enjoy the digital world, which brings us possibilities that we have never before contemplated as a species. But the Law of Unintended Consequences still applies: There is a school of thought, and a thousand Hollywood movies heavily CGI-d, that we have never made contact with aliens because, in advanced technological civilisations, the ability for intelligent life to control its tools is eventually surpassed by the capability of those tools to act autonomously. That once a civilisation has the power to destroy itself, it is only a matter of time before that power is accidentally or deliberately used. Just as they were able to talk to us, those aliens blew themselves up, and perhaps that version of the omniverse explains why there's been no sign of aliens in the 80 years since we invented radio receivers and have been able to detect them.

While humans are not the only creatures who use tools, we have over the past 10,000 years taken them to levels exponentially beyond those of our planet's fellow species. The relationship between ourselves and our computers large and small, local or dispersed, is at the heart of our generation's success. Yes, if we abuse that relationship the digital era may bite us, let's hope nothing worse, no asteroid to wipe us out. But used intelligently and in partnership, it offers the key to our future development. The speed and force of the change in that partnership turns it into a moment in the evolution of our species of Darwinian significance to us all.

Chris Wise
April 2020

Foreword

I joined the structural engineering profession forty years ago when computer analysis was just taking hold as a mainstream design activity. My whole career has been built upon trying to use new digital techniques as and when they become available. In general, my observation is that their use is more limited by our lack of imagination and curiosity, combined with a natural risk aversion common among engineers, than it has been by the technology itself.

Over that period, structural analysis, or performance simulation, has not changed very much but our attitude towards it has. We used to use it to prove ideas and designs that we had already fully conceived. It simply checked the sums and made sure that we had not made a numerical mistake.

Nowadays we use simulation to explore conceptual possibilities: what if? How might this structure compare with that one? Which is more carbon efficient (cost efficient, material efficient…) while being easier to build and providing the best human experience? These are examples of the sort of questions that we increasingly use digital simulations, building information models, visualisations and virtual reality to help answer.

At all times, simulation models have the capability of being double-edged swords. We still need to use our engineering judgement and experience to decide what to model and how to represent it. When I was first taught about finite element analysis (FEA), I was told that such an analysis would converge on a good solution only as you continuously refined the mesh. From that, I have learned to always build lots of different models; simple ones and complicated ones; two dimensional and three dimensional; of components, assemblies and of the whole structure. I do not agree with those that assert that you need to be able to check the model by hand, but I do agree that all models need to be checked by making lots of other models — ideally by different people using different software.

In the immediate future we can start harnessing Machine Learning (ML) and Artificial Intelligence (AI) to better assess the outcomes of these simulations. Once we have processed a whole set of parametric simulations in the cloud, we can unleash ML/AI to sift and sort the results, and extrapolate towards more fertile possibilities.

We can use 3D printing to test the forthcoming ideas at model scale and additive (and subtractive) manufacturing to test them at full scale; we can simulate all aspects of their real-world performance, such as auralisation, before committing to a physical reality. In the truly virtual world, we can simulate millions or billions of possibilities and not care at all if 99.999% of them fail, provided we find the single solution that best meets the project needs — such profligacy does not always sit comfortably with engineers.

These tools and techniques will result in a more delightful and efficient built environment. But what are the opportunities for then taking a further step and harnessing “internet of things” devices to monitor real world performance? We could reduce our design loads and material factors of safety if we can reliably monitor real-world loads and real-material performance.

We are also likely to see other branches of technology, for example biotechnology and nanotechnology, help us in our search for more carbon-friendly materials that are also robust and durable.

Right now, we have the ability to halve our usage of new materials, to reuse more existing structures, to increase recycling and to minimise waste, while improving the world around us and regenerating ecosystems — just by using the computational techniques currently available and well described by Peter Debney in this book.

Tristram Carfrae
May 2020

Foreword

Every generation of structural engineers goes through a combination of evolutionary and revolutionary change throughout their career. We are always learning; through formal education, our career experience, sharing knowledge and our observations through life. But we often become very set in our ways, with fixed ideas of what is right and wrong.

The potential of computational design and the rapidly expanding role of digital technology throughout our lives and workplaces still invokes fear in many, and it is essential that we do not blindly use the powerful tools we now have at our disposal without understanding the fundamental engineering principles behind them.

Not all of us will be software developers or coding specialists, but when we use these tools it is essential that we at least understand what is going on inside the "black box". Using Peter's eloquent analogy in this book, knowing the mechanics of an engine does not make you an experienced driver, and *vice versa*. But as the driver it is essential to understand how to control the vehicle and safely harness its power, so you can trust the relationship between human and machine.

Our understanding of structural materials, new and old, and of simulating and modelling structural behaviour continues to expand. However, in terms of the first principles of mechanics and mathematics, the algorithms and calculations we are working with day to day do not fundamentally change. Computers can do more, faster, but they are still only tools and as engineers we are still the craftspeople. Tools can speed up common and repetitive tasks, but in the right hands they can also create beauty and innovation.

So, by embracing a more balanced partnership between the engineer and digital technology, we can use computational design not only to automate the mundane tasks but to unleash our creativity. Using generative design and multi-objective optimisation algorithms, we can explore and objectively assess a myriad of efficient design options that our experience-biased minds might not have considered in the first place. And using Machine Learning to make sense of the huge amounts of data at our fingertips we can place ourselves in a much more central and active role in the design process — something we can all aspire to as professionals.

This book provides an enjoyable and holistic introduction to many of these concepts — serving as a stepping stone for those embarking on the early part of their engineering career, but also as a very useful point of reference for those of us more set in our ways, making us critically assess our philosophy as designers, our imperfect relationship with technology, and our understanding of why things sometimes still go wrong!

Jon Leach
April 2020

Peter Debney

Arup/Oasys Software

Since gaining my degree in Civil Engineering at Surrey University over 30 years ago I have designed sewage treatment plants and petrochemical refineries, houses and shopping centres, portal frames and reservoirs. I have worked as a CAD technician and as an engineer, creating drawings, calculations and macros. I have also worked directly in the engineering software industry, helping develop the CAD programs *Xsteel* (*Tekla Structure*) and *3D+*, and the finite element analysis and design program *GSA*.

Computing has dominated my life and career; from trying (and failing) to write games for the Sinclair ZX81, via using analysis and CAD programs to design buildings, to designing and supporting those programs for others to use, I have seen every side of the life of high-tech structural design. I have also been teaching those programs and the engineering behind them for more than 20 years.

I am also conscious that, though it began over 50 years ago, we are still at the start of the engineering computing revolution. We can now achieve things that were not possible only a decade ago, but are also in danger of forgetting important lessons of the past. Most importantly, the climate crisis means that we must do more than design structures that work: we must do more with less. Optioneering and optimisation are now key tasks for the modern engineer, and the computer is the tool we need to help us fulfil them.

Acknowledgements

Permission to reproduce the following has been obtained, courtesy of these individuals/organisations:

Cover *The Lattice Ceiling in Kings Cross Station* © chrisdorney — stock.adobe.com
Author image © Beatrice Debney
Figures 1.1 and 6.17 © Beatrice Debney
Figure 1.2 © Tim Ibell
Figure 2.1 © Daniel Imade_Arup
Figures 2.2, 2.5, 3.12, 3.13 and 8.9 © Arup
Figure 2.4 © Seagate Mass Timber Inc./Pollux Chung
Figure 2.7 © Benh LIEU SONG (CC BY-SA 3.0)
Figure 2.8a © George Gastin (CC BY-SA 3.0)
Figure 2.8b © Simon Johnston (CC BY-SA 2.0)
Figures 3.1, 3.23, 9.12, 9.16 and 10.14 © xkcd https://creativecommons.org/licenses/by-nc/2.5/
Figure 3.3 © Paul Carstairs
Figure 3.4 © Nigel Whale_Arup
Figure 3.5 © Rasmus Hjortshoj_COAST
Figure 3.9 © Grant Smith_VIEW
Figure 3.11 © Hufton+Crow
Figure 3.18 © Robert Leighton
Figures 3.19–3.22 © The Institution of Structural Engineers
Figure 6.22 derived/adapted from Figure 3.1 of BS EN 1998-1 (BSI)
Figure 7.1 Jerry Williams. Copyright © 2020. Hartford Courant. Used with permission
Figure 7.2a © Bair175 (CC BY-SA 3.0)
Figure 7.2b © Bluemoose (GNU free documentation licence)
Figure 7.3 © Mark Mitchell, The New Zealand Herald
Figure 7.32 © Transport Scotland
Figure 7.41 © Vincent Ramet
Figure 8.5 Courtesy of © SOM
Figure 8.6 © MAXXI Museo nazionale delle arti del XXI secolo, Rome. MAXXI Architettura Collection (F5506)
Figure 8.8 © David de Jong_Arup
Figure 8.10 (top) © Emerald Publishing Limited
Figure 8.30 © ETH-Bibliothek Zurich, Bildarchiv (CC-BY-SA 4.0)
Figure 8.32 ETH Zurich, Block Research Group © Iwan Baan
Figure 9.11 derived/adapted from Dobbyn *et al*.[146]
Figure 9.13 © Johann Dréo (GNU free documentation licence)
Figure I1 © David Brohn

Others:
Figures 3.16, 7.31 and 10.15 public domain
Figure 3.17 US Army (public domain)
Figures 4.7, 6.32, 6.36, 6.40, 6.45 and 6.46 derived/adapted from Hellen and Becker[53]
Figure 4.14 derived/adapted from *Guidelines for the use of computers for engineering calculations* (IStructE Ltd)
Figure 5.4 US Government (public domain)
Figure 6.44 derived/adapted from *Train Test Crash 1984 — Nuclear Flask Test*[81]
Figures 6.47 and 8.33 derived/adapted from Oasys GSA training material
Figure 7.20 derived/adapted from Martin and Delatte[85]
Figures 7.21, 7.22 and 7.23 derived/adapted from Schlaich and Reineck[88]
Figure 7.40 NZ Government (Crown Copyright)
Figure 8.2 derived/adapted from Michell[108]
Figure 8.10 (bottom) derived/adapted from Fairclough *et al*.[114]
Figure 8.16 derived/adapted from Allen and Zalewski[116]
Figure 10.1 Eadweard Muybridge (public domain)
Figures 10.16 and 10.17 © M.Bongard/Macmillan Publishers
Figure 10.18 derived/adapted from 'Fibonacci' (Wikipedia) (CC BY-SA 3.0)
Figure 11.5 derived/adapted from *Understanding Quantum Computers*[230]

Permission to reproduce extracts from British Standards is granted by BSI Standards Limited (BSI). No other use of this material is permitted. British Standards can be obtained in PDF or hard copy formats from the BSI online shop: www.bsigroup/Shop

Preface

"A lot [of structural engineers] are acting as human calculators and that's not engineering. If that is what people are doing, they will soon be replaced by computers, and that's a good thing. Because then they will be free to do what humans do best: complex problem solving; dealing with new phenomena; and being human."
Chris Wise[1]

The engineer and the computer

In my first year at university I dutifully attended the maths classes. Although I had been top of my school in maths at 16, I disliked A-level calculus and despaired to learn that I would need to do a lot more of it, both at university and in my career. It was a relief then, when I went into the subsequent structures class and was told that we could ignore calculus; instead there were plenty of standard formulas that we would use to calculate the results. This, I thought, was more like it. Finally, I attended the computing class, where we were told that we did not even need to know the standard formulae but could just put the structure into the computer and let it do all the calculations. This was the answer I was looking for!

Of course, as I subsequently learned, these statements were not entirely correct. Rather, like so much in engineering, they were 'near enough'. I have subsequently found occasional uses for calculus, and standard formulas can be incredibly useful. Where would we be without $M = wL^2/8$ for example: it is the structural engineer's equivalent to $E = mc^2$! Such a simple formula yet it allows us to rapidly analyse not only beams, but also trusses, arches and catenaries, if you know how to apply it.

But it was to the computer that I kept returning, whether for creating computer-aided drawings or finite element analysis models, many of which would be extremely difficult, if not impossible, to design with paper and pencil.

I also discovered that computers are not the answer to every problem and, like the oracles of old, their answers need the most careful scrutiny and thought. They will, usually, give you an answer, but it is for you to ensure that you asked the right question and received the right answer. The more experienced you are as an engineer the better you become at judging the results of computer calculations. "The purpose of computing is insight, not numbers" said Richard Hamming back in 1962 and he was, in many respects, correct, though numbers are still particularly important to the engineer[2]. I have learned a lot from building computer models of structures to see how they behave, but usually the models are of structures that I already understand, so I use them to extract the value of the forces and moments.

As a graduate I focused a lot on engineering computing, so it came as quite a shock when I started preparation for the IStructE Chartered Membership exam: computers were not allowed. But as my preparations continued, I realised that this was quite correct. To be a chartered structural engineer you must know the answers before putting pen to paper, or mouse to computer. Also, you cannot analyse a structure until you've designed it, and the design needs to be decided by you. Computers are an excellent tool for engineers to fine-tune their designs: check exactly what section sizes are required, and perhaps tweak the geometry to maximise efficiency. They also help when it comes to revising the design when the architect has, invariably, changed their mind again.

Computer programs are not universal panaceas: they will help a good engineer produce better designs and a bad engineer produce worse ones. They are only tools to magnify our abilities. There is no substitute for engineering skill, but the best engineers also make the most of computers.

Workshop manual or driving handbook?

Computers have enabled the modern engineer to create more efficient, more elegant, and more sustainable structures faster than they ever could have done using the methods of previous generations. While we may know less now about advanced mathematical and empirical methods than our forebears, who had no choice in the matter, we instead need to know more about the electronic computer, the programs that run on them, the methods they use, and how to use them all to best advantage. Among everything else that project work demands, today's graduate engineer must also be knowledgeable in computational engineering. So how is this knowledge acquired?

One of the primary tools of the modern computational engineer is finite element analysis (FEA), which allows us to analyse almost any structure in multiple ways. Most universities and textbooks approach FEA by teaching about its inner workings in detail but then ignore how to use it. This is like teaching someone how the internal combustion engine works, then after passing a written exam on the advantages of fuel injection over a carburettor and calculating torque from a diesel motor, awarding them a driving licence. We would not let anyone drive a car unless they had proved their competence. Yet junior engineers are often just as inexperienced in 'driving' engineering software.

So, what does a driver need to know? Do not put diesel into a petrol engine and *vice versa*; keep the oil and water topped up; and get the car serviced regularly. You also need to know which side of the road to drive on, how to behave at junctions, and how to stop in an emergency. Driving an engineering computer program is just the same. You need to know how to build good models and avoid bad ones, to let the computer deal with the repetitive and mundane tasks, and to use the computer to explore efficiencies and possibilities in design that were just not possible a generation ago. As practicing engineers, we need to have a working knowledge of FEA even if we are not experts, just as we need to know something about materials, construction methods, fire, architecture, and other problems that we must deal with on a typical design project.

There are many excellent textbooks available on the detailed workings of FEA, on optimisation methods, on the detail of the Industry Foundation Classes that allow communication between engineering programs, and so on. These are all invaluable if you are going to write your own programs or want to deepen your understanding of how those programs work. If the chapters in this book ignite your interest, then do seek them out. Likewise, talk to the highly skilled engineers and mathematicians who teach FEA modules and write programs — they've spent years studying engineering computing in detail. Like all engineering experts they are a great resource for those occasions where we need to dig deeper into a project problem. Meanwhile this guidance will introduce you to the many aspects of computing for structural engineers, bringing them together to look at the whole design process, with a minimum of maths but a maximum of explanation and context.

Computational engineering

The structural engineering profession is on the verge of a new generation of technology and innovation: the fourth industrial revolution. The 1960s and 70s gave us the mainframe and the first finite element analysis; in the 1980s CAD became mainstream and personal computers (or PCs) replaced the behemoths; mobile phones in the 1990s lead to the mobile technology revolution and smart phones of the new century; and more recently the industry (finally) woke up to the communication and workflow possibilities afforded by BIM. Now we are beginning to see artificial intelligence (AI) and machine learning (ML) applied to everyday engineering tasks.

Artificial intelligence has been an ambition of computing since its very beginnings with Alan Turing's 1950 paper *Computing Machinery and Intelligence*[3]. AI has had its setbacks, and while it has not achieved the original goal of creating a general intelligence, certain aspects have been phenomenally successful. Computing power, especially the parallelisation possible with distributed cloud computing, is now making optimisation a working proposition for everyday engineering. 3D printing, robotics, and zero-carbon construction means that very soon material usage will dominate over labour costs, necessitating the application of optimisation techniques that our mechanical engineering colleagues have been using for some years. And the computational power of modern engineering computer programs means that the traditional tasks of sizing steel members and rebar quantities are now regularly automated.

This all means that the time has come for a fresh look at computing for structural engineers and what our role is in this age of machine learning and automation. What does the graduate of today actually need to know and how can they compete? The engineers of today must embrace computers as digital engineering assistants if they are not to replace us. We must use their strengths and know their limitations, let them free us up to do that what computers can (probably) never do — understand the client's needs — then use imagination and ingenuity to solve their problems.

So, what do engineers need to know about these computer programs? As always it is best to start at your goals and work backwards. Think of it like those puzzles we had as children, where there are several staring points and some end points, joined by lines resembling a plate of spaghetti. I realised quite young that the way to solve those puzzles was not to try the start points, but to instead start at the end and work back. Sometimes trial and error can

be a useful thing though, as we will explore in the chapter on optimisation: answers can sometimes depend on where you start from. There is rarely only one ‘correct’ answer.

Apart from computing, this book is about design, but that also raises the question: ‘What is design?’ If you ask any civil engineering student, they might tell you that design is determining the size of the steel beam or the reinforcement needed in a concrete beam using a design code. But these are both trivial problems. You put the forces and moments into the formula, turn the handle, and the answer drops out. In the case of a steel beam you may then need to choose a section that meets the requirements from a table, but little more than that. It is just a mechanical process with a single answer. Or is it that simple?

It is easy to find the lightest steel section, but is it the cheapest or the one with the lowest environmental impact? What happens when we take the connections into account, is it still the best, or are we better-off using a larger section and make the connections simpler? Should we make the section the same as some others to increase manufacturing efficiency? But clearly making all the beams on the project the same depth, whether they are spanning 1m or 10m, is not going to be efficient — so what is the answer?

If you are just focusing on optimising the section sizes you are missing the bigger picture: should the member even be there, or should it be somewhere else? You may tune the chord sizes on a truss but varying the depth of the truss will have a far greater impact on the end result; but did you test that or just use a standard span-to-depth ratio and make it work? What if you change the number of diagonals in the truss, will that make it better or worse? Does the bottom or top of the truss really need to be straight, or would a different shape be better? Should it be a Warren, a Fink, a Prat, or even a Michell truss? We have not seen many Michell trusses so far, but I expect that they will be far more popular in the future. What if the columns were further apart, or closer together? What if the beams spanned the other way? What if we change the roof slope or even invert it? Should we use steel, concrete, timber, or consider an alternative material?

The lightest design is not the cheapest design is not the design with the lowest environmental impact is not the design with the shortest time on site is not the design with the simplest construction. Usually. Design codes do not tell us how to design, despite their name, but just give us the basics: strengths, formulas, minimum requirements, and so on. But how do we find the best design?

“Engineering problems are under-defined, there are many solutions, good, bad and indifferent. The art is to arrive at a good solution. This is a creative activity, involving imagination, intuition and deliberate choice.”
Ove Arup

Despite our training in maths and physics, where we are taught that there is one answer, design is not so simple as we may have been led to believe. As some of my colleagues would put it: it is “non-trivial”. Personally, I would say that the challenge makes design more interesting. How do we produce good designs when there are so many variables and no single answer? It takes a long time for us to check a single design. We cannot check them all, but this is where computers can help us.

One advantage of computers is that they are incredibly good at doing the maths for us — the clue is in the name. Another is that they do the maths very quickly. And a third is that they do not tire or complain about doing the same thing over and over again. While we, if working with pencil and paper, might choose just one design, we can tell the computer to test 100, then take the best and from those make 100 more. Or we might ask it to keep adjusting the position of a connection until it has found the best location. Or we might ask it to examine all the designs that we have done in the past and suggest what the best answer is likely to be this time.

As a young structural engineering graduate, I started in the industry in the early days of the computer analysis and drafting revolution. Now we see the same happening to design, just at the time when design, especially the environmental impact of our designs, is becoming so important. We have the responsibility and we have the tools at hand to make this world a better place, if we know how to use the tools wisely. Let’s get started.