1

The Computational Engineering Sciences: an introduction

1.1 Engineering Simulation

The digital computer, coupled with engineering and computer science plus modern approximation theory, have coalesced to render computational simulation via math modeling an alternative modality supporting *design optimization* in engineering. Design has historically been conducted in the *physical laboratory* (Figure 1.1). The test device is a miniature of reality and the laboratory process sequence is:

- *model* the geometry (similitude)
- *determine* desired data (cost)
- *acquire* the data
- *interpret* the data
- *draw* conclusions

The *computational engineering sciences laboratory* has emerged as the complement to, or replacement of, the legacy modality (Figure 1.2). The computational laboratory process sequence is:

- *model* the mathematics (fidelity)
- *model* the physics (cost)
- *compute* the data
- *interpret* the data
- *draw* conclusions

Finite Elements ⇔ Computational Engineering Sciences, First Edition. A. J. Baker.

Figure 1.1 Classic wind tunnel test

The first two components of the computational engineering sciences (CES) laboratory place a significant *new burden* on the engineer/scientist. Aspects of calculus and vector field theory, the *language* for expressing conservation principles in the engineering sciences, must be recalled. Additionally, dexterity with *constitutive closure* approximations, that is, the "*physics model*," must be understood on a fidelity/mathematics as well as cost/benefit basis.

The identical calculus and vector field topics underpin modern approximation theory guidance for generating a conservation principle *approximate solution* based on a *weak formulation* (WF) [1]. The mathematicians, in developing this approach to solution approximation, have endowed it with an elegant theory on *optimal* construction and error estimation. A WF, *always* completed in the *continuum*, theoretically *transforms* the

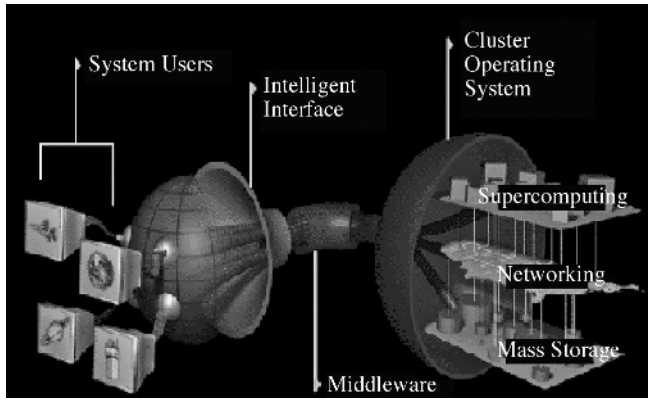

Figure 1.2 Cloud-computing visualization

solution of the partial differential equation (PDE) into a *computable* large-order algebraic equation system.

Once the continuum weak form theory is completed, the sole remaining decision is implementation. Herein this is accomplished by replacing the trial and test spaces with *finite element* (FE) *trial/test space bases* defined for a spatial *discretization* of the PDE domain of dependence. This identification directly enables WF integral evaluations using the calculus. Detailing this process for a diverse spectrum in the *engineering sciences* is the content of this text.

1.2 A Problem-Solving Environment

Historically, a frustrating aspect of computational simulation was interacting with *the computer code*! User interface and computer science issues dominate this facet, and the engineer/scientist interested in analysis is typically not well founded in the required skills. This issue is compounded by the tradition in *olden times*, that is, a decade or so ago, for the individual to code his/her own computer program.

This incredible dissipation of time and effort has been superceded by the emergence of *component-based software* leading to the concept of a *problem-solving environment* (PSE). Commercial code systems now exist throughout the engineering sciences possessing very powerful advances in user interfaces. Maturation of *grid computing* concepts will lead, in the not too distant future, to Internet-enabled *just-in-time* capabilities using remotely accessible *high-performance computing and communications* (HPCC) constructs [2].

Figure 1.3 illustrates this emergent scenario. The practicing design engineer possesses knowledge about his/her problem statement, and after absorbing this text's content will be thoroughly comfortable with the associated mathematics/physics issues with seeking an *optimal* approximate solution. From that point on only casual

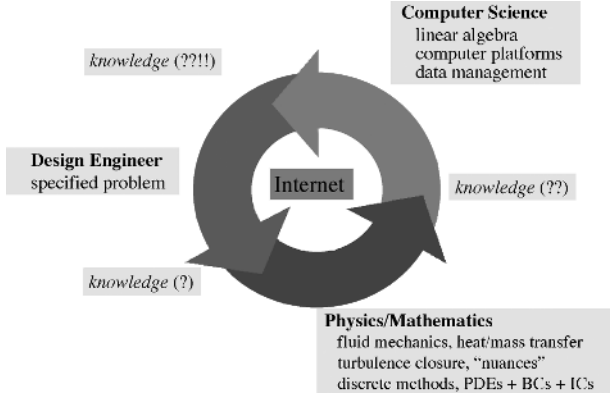

Figure 1.3 The problem-solving environment

knowledge about the subsequent computer science issues will be required, as the Internet modality exists to complete the loop.

An added historical aspect is that computational codes were irrevocably tailored to the specific discrete theory, for example, finite difference (FD), finite volume (FV), FE for a given engineering science problem class. This is now moot as completed research confirms these apparently very distinct computational constructs can be interpreted as specific decisions in implementing a *weak statement* (WS), the *extremum* of a WF for a PDE. Invariably the FE discrete implementation generates the *optimal* construction, the consequence of WF theory, and the use of calculus rather than difference algebra to form the algebraic statement.

The computational practice of FE methods is rapidly maturing, as academics in math, engineering, and computer science collectively resolve key theoretical issues. A by-product, developed in thoroughness in this text, is the *object-oriented* FE algorithm construct that directly communicates "compute desire" to a PSE via a *template*.

This approach recognizes a code is but a data-handling system, and the FE implementation of a WS generates *only* six data types for each and every (!) FE domain Ω_e specifically including nonlinearity. The objects for *all* element-level matrix contributions $\{\text{WS}\}_e$ to a WS algebraic statement are thus organized as:

$$\{\text{WS}\}_e \equiv \begin{pmatrix} \text{global} \\ \text{constant} \end{pmatrix} \begin{pmatrix} \text{element} \\ \text{average} \end{pmatrix}_e \begin{Bmatrix} \text{element} \\ \text{variable} \end{Bmatrix}_e \begin{pmatrix} \text{metric} \\ \text{data} \end{pmatrix}_e \begin{bmatrix} \text{master} \\ \text{matrix} \end{bmatrix} \begin{Bmatrix} \text{unknown} \\ \text{or data} \end{Bmatrix}_e .$$

Coding of a FE WS discrete implementation is thus reduced to data identification in these six object categories.

Herein, the progression of a WS algorithm for an engineering science topic, FE discrete-implemented, leads to the object-oriented *template* transparently converting theory to executable code. Template generation occurs in a word-processing environment, and the result precisely encompasses all complexities, specifically including nonlinearity, in coupled PDE systems. The template-enabled computing PSE herein employs MATLAB® [3], via the specifically written *FE*m*PSE toolbox* for expository computing labs. Design-based computing experiments employ COMSOL [4], an FE-implemented *multiphysics* commercial PSE.

1.3 Weak Formulation Essence

An engineering design problem statement is invariably cast as a PDE written on the *state variable* (the dependent variable), herein labeled $q = q(\mathbf{x})$ for the steady definition. The compact notation used in this text to denote a PDE is

$$\mathcal{L}(q) = 0, \text{ on } \Omega \subset \Re^n. \tag{1.1}$$

In equation (1.1), $\mathcal{L}$ is the PDE placeholder and its domain of influence is symbolized as Ω, a region lying on an n-dimensional euclidean space $\Re^n$.

To "connect" the PDE to the specific problem statement requires boundary conditions (BCs) communicating this given information, that is, the *data*. The text-utilized BC compact notation is

$$\ell(q) = 0, \text{ on } \Omega \subset \Re^{n-1}, \tag{1.2}$$

where $\partial\Omega$ is the (n–1)-dimensional bounding enclosure of Ω. Figure 1.4 illustrates these formalisms.

The exact solution $q(\mathbf{x})$ satisfying a *genuine* problem equations (1.1) and (1.2) can never (!) be found analytically. Consequently, the *key* WF theory requirement is to formally define an (any!) *approximation* to $q(\mathbf{x})$. Herein this requirement is expressed as

$$q(\mathbf{x}) \approx q^N(\mathbf{x}) \equiv \sum_{\alpha=1}^{N} \Psi_\alpha(\mathbf{x}) Q_\alpha. \tag{1.3}$$

The assumption in equation (1.3) is that one can identify a suitable *trial space* $\Psi_\alpha(\mathbf{x})$, a set of functions on $\Re^n$, to *support* any approximate solution. The summation therein couples each trial space member to an *unknown* expansion coefficient Q_α, called a *degree-of-freedom* (DOF) of the approximation, the set of which is to become determined in the algebraic computing process.

Unless equations (1.1) and (1.2) define a trivial problem, q^N cannot be identical with q. The difference between q and q^N is the *approximation error*, herein denoted e^N. Since everything is a function, obviously

$$e^N(\mathbf{x}) \equiv q(\mathbf{x}) - q^N(\mathbf{x}). \tag{1.4}$$

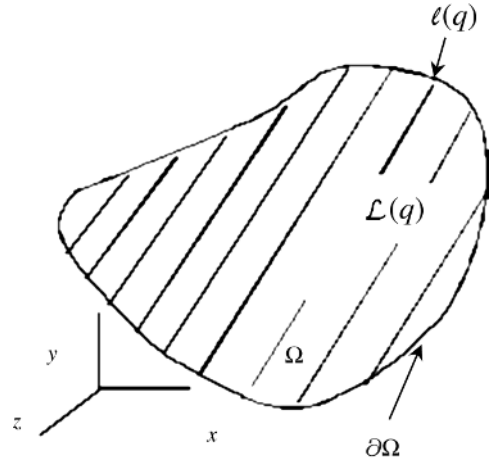

Figure 1.4 Engineering problem statement notation

The singular goal is to seek the *best* approximation q^N, hence to *constrain* in some sense the "size" of $e^N(\mathbf{x})$. This is elegantly accomplished via the mathematicians' *WF* that requires the available measure of error

$$\mathcal{L}(q^N) \neq 0\,!$$

be made *orthogonal* (mathematically "perpendicular") to an *arbitrarily* chosen *test function* $w(\mathbf{x})$. The *weak form* (WF^N) expression of this constraint on the approximation (1.3) is

$$\mathrm{WF}^N \equiv \int_\Omega w(\mathbf{x})\mathscr{L}(q^N)\mathrm{d}\tau \equiv 0. \tag{1.5}$$

The requirement of *any* function $w(\mathbf{x})$ is cleanly handled via an interpolation, ([5], Chapter 2.2), followed by forming the *extremum* of rearranged equation (1.5) which identifies the *test space* $\Phi_\beta(\mathbf{x})$ companion to the trial space $\Psi_\alpha(\mathbf{x})$. The scalar equation (1.5) thus becomes a *set* of equations of the order N defined in equation (1.3), which is termed the *weak statement* (WS^N). All $\mathbf{x}$-dependence vanishes in evaluating the defined integrals, hence WS^N generates the *algebraic* equation system

$$\mathrm{WS}^N \Rightarrow [\mathrm{Matrix}]\{Q\} = \{\mathrm{b}\}. \tag{1.6}$$

As the final caveat, inserting equation (1.2) into equation (1.5) moves the BC-constrained DOF in the set Q_α, into the *data* matrix {b} in equation (1.6). The remaining equation (1.2)-defined DOF populate the column matrix $\{Q\}$ in equation (1.6), the *exactly* correct order algebraic equation system for determination of the *unknown* DOF defined in equation (1.3).

1.4 Decisions on Forming WS^N

The key to weak form utility is the *assumption* that the integrals in WS^N, equation (1.6), can be evaluated. This obviously centers on the functional form selected for the test and trial function spaces. These decisions in turn identify a specific algorithm from the wide range of WS^N methods that can be derived. The following table provides a WS^N summary *essence* categorized on these function sets.

$\Phi_\beta(\mathbf{x})$,$\Psi_\alpha(\mathbf{x})$	Examples	WS^N Label
Global	Sine, cosine, Bessel, spherical harmonics	Analytical methodology (separation of variables)
	Chebyshev polynomials	Spectral methods
Global–local	Chebyshev by blocks	Pseudospectral methods
Local	Lagrange polynomials	Spatially discrete methods (FE, FV, FD)

For $\Phi_\beta(\mathbf{x})$ and $\Psi_\alpha(\mathbf{x})$ spanning the entirety of Ω generates formulations closely associated with analytical PDE-solution methodology. However, this choice precludes geometric flexibility, as closures $\partial\Omega$ of the domain Ω must be coordinate surfaces. The singular key attribute is that these spaces contain functions that are indeed orthogonal on Ω. Hence, [Matrix] in equation (1.6) is typically diagonal, which renders the algebraic solution process trivial (recall *separation of variables* in your sophomore calculus class?).

Spectral methodology retains the definition and use of global span function spaces. Pseudospectral methods lie halfway between spectral and spatially discrete algorithms, and both typically inherit the liability that closure segments be coincident with global coordinate surfaces.

For domains with absolutely arbitrarily geometric closure, that is, essentially *all* practical problems, the FE discrete implementation of WS^N, hereon denoted WS^h, *guarantees* the extremum of WF^N, equation (1.5), generates integrals that can be evaluated. This is accomplished by subdividing Ω into the *union* (nonoverlapping sum, symbol $\cup$) of small subdomains, see Figure 1.5. Each subdomain is called a FE, denoted Ω_e, and their union can be manipulated to fit any geometrical shape of the domain closure $\partial\Omega$.

A WS^N can be manipulated to interpret FD and FV methodology as will be illustrated. The formulation distinctions include integrals not being generated via calculus and the resultant algorithms are not predictable *optimal*, the key attribute of a specific FE implementation WS^N, detailed shortly.

This process of subdividing Ω into the *union* of small subdomains is called spatial *discretization*, symbolized in the literature by superscript *h*. Unambiguously then

$$\Omega \approx \Omega^h \equiv \cup_e \Omega_e \tag{1.7}$$

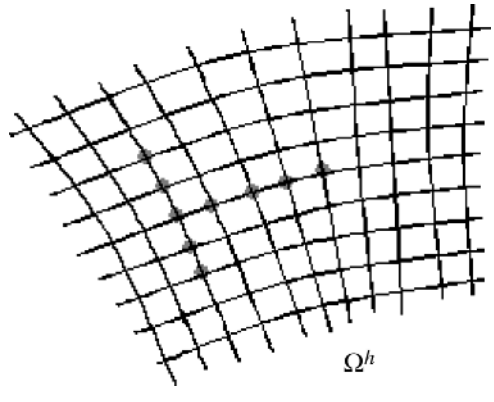

Figure 1.5 Domain discretization

and the region Ω_e is called a *FE*. The resultant FE solution approximation form (1.3) transitions to

$$q^N(\mathbf{x}) \equiv q^h(\mathbf{x}) = \cup_e q_e(\mathbf{x})\}. \tag{1.8}$$

Constructing the required discrete equivalents of $\Psi_\alpha(\mathbf{x})$ and $\Phi_\beta(\mathbf{x})$, equations (1.3) and (1.5), generates the trial and test space *basis functions*. The theoretical foundation is typically Lagrange or Hermite interpolation polynomials, and FE basis functions are herein symbolized as the column matrix $\{N(\mathbf{x})\}$. Hence, for equation (1.8)

$$q_e(x) = \{N(x)\}^T \{Q\}_e. \tag{1.9}$$

With equations (1.7)–(1.9) the WS^h-generated [Matrix] in equation (1.6) is *never* diagonal. Hence, one must find an algebraic solution replacement for Cramer's rule, which introduces iterative matrix *linear algebra* methodology.

A *fundamentally significant* solution facet results upon making the discrete approximation decision. In the FE implementation, the DOF $\{Q\}_e$ in the element-level approximation equation (1.9), that is, select DOF Q_α in equation (1.3), are usually generated *only* at mesh intersections on Ω^h. Illustrated in Figure 1.5 as dots (•) they are called the *nodes* of the mesh.

The union of the local solutions $q_e(\mathbf{x})$ forms $q^h(\mathbf{x})$, equation (1.8), with the resultant *spectral resolution* controlled by node-separation distance. Specifically, for a mesh of *measure* Δx any $2\Delta x$ wavelength information *cannot* be resolved. This is clearly illustrated in Figure 1.6; on the left the DOF $\{Q\}$ for the $2\Delta x$ sine waves are all zero (!) while those on the right are nonzero. Hence *mesh resolution* is central to accuracy; a too coarse mesh can produce totally wrong solutions, as will be illustrated.

1.5 Discrete WS^h Implementations

Legacy FD and FV methods also employ a domain *discretization* $\Omega^h = \cup_c \Omega_c$, where Ω_c is a computational *cell*. Further, mathematicians and chemical engineers (in particular) have developed many node-based numerical methods, for example, *collocation*, *least squares*, *weighted residuals*. Do fundamental theory underpinnings exist for these apparently very diverse discrete procedures for PDEs?

The answer is a resounding *YES!!* Under the weak form umbrella, the distinctions reside strictly in the test and trial space *basis functions* chosen to form WS^h. The following table summarizes algorithm decisions in the context of WS^h.

Name	Trial space, $\Psi_\alpha(\mathbf{x})$	Test space, $\Phi_\beta(\mathbf{x})$
Galerkin (FE)	Basis $\{N\}$	Basis $\{N\}$
Collocation	Basis $\{N\}$	Kronecker δ
Finite difference (FD)	?	None
Finite volume (FV)	?	Unity
Least squares	Basis $\{N\}$	$\mathcal{L}$ $(\{N\})$
Boundary element (BEM)	Basis $\{N\}$	Green's function

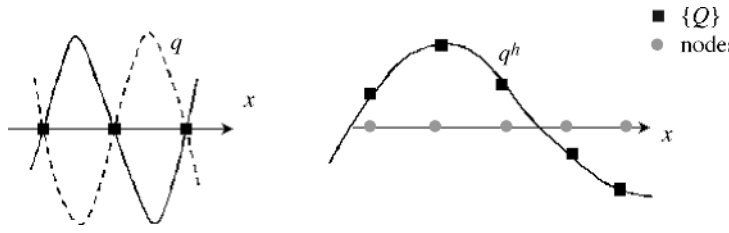

Figure 1.6 Resolution illustrations on a mesh of measure Δx

The fact that myriad choices exist, and have been computer implemented, immediately raises the *fundamental* question:

Does an optimal choice for the WS^N trial and test space function sets $\Psi_\alpha(\mathbf{x})$ and $\Phi_\beta(\mathbf{x})$ exist?

One must first define *optimal* to answer this. Mathematicians will work this to the point of distraction but engineers are not so burdened. Their obvious choice is *the* selection that produces the absolute *minimum* approximation error $e^N(\mathbf{x})$, equation (1.4).

Importantly, this answer must be and *is* absolutely independent of the particular choice for discrete implementation! For a wide range of PDEs describing problem statements in the engineering sciences, in the *continuum* the answer to the fundamental question is:

The WS^N approximation error $e^N(\mathbf{x})$ is minimized, in a suitable norm, when the trial and test spaces are identical.

Now moving to the WS^N spatially discrete implementation WS^h, on a mesh Ω^h, the approximation error becomes $e^h(\mathbf{x}) \equiv q(\mathbf{x}) - q^h(\mathbf{x})$. Thereby, the WS *discrete* implementation corollary for optimal performance is:

The WS^h approximation error $e^h(\mathbf{x})$ is minimized, in a suitable norm, when the trial and test space replacements contain the identical FE trial space basis functions.

The name historically attached to identical trial and test space functions is *Galerkin*; herein this form of WS^N is denoted GWS^N. The Galerkin criterion is *optimal in theory and in the FE discrete implementation*.

The vector "cartoon" in Figure 1.7 serves to illustrate that the exact solution q cannot lie in the "plane" containing the trial function set $\Psi_\alpha(\mathbf{x})$ supporting q^N unless they are identical (not likely!). The "distance" between q and q^N is the error e^N and its "magnitude" is the smallest when e^N is *orthogonal* (mathematically perpendicular) to the plane, as induced by the Galerkin criterion $q^N(GWS^N)$. The solution $q^N(WS^N)$ generated by any other trial/test function criterion produces the error $\bar{e}^N$ which is not

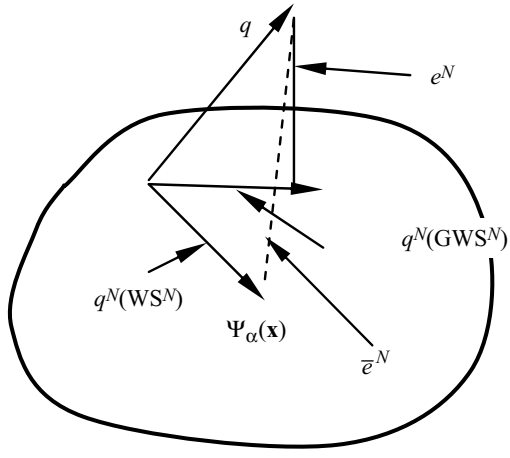

Figure 1.7 Vector cartoon illustrating GWS^N optimality

orthogonal to the trial/test function plane, hence the (dashed) error vector $\overline{e}^N$ possesses a larger "magnitude."

1.6 Chapter Summary

The FE discrete implementation of $\mathrm{GWS}^{\boldsymbol{N}}$, that is, $\mathrm{GWS}^{\boldsymbol{h}}$, is the *optimal* decision for a wide class of problem statements in the CES. FE trial space basis availability guarantees accurate evaluation of the integrals defined in equation (1.5) for domains Ω enclosed by an arbitrarily nonregular shaped boundary $\partial\Omega$. With this theory in place, the key topic becomes identification of the *trial space basis functions* spanning FE domains Ω_e on n-dimensions. Their identification and performance quantification across the *CES* is the subject of this text.

In summary, the $\mathrm{GWS}^N \Rightarrow \mathrm{GWS}^{\boldsymbol{h}}$ process *essence* for designing optimally performing algorithms is:

$$\text{approximation: } q(\mathbf{x}) \cong q^N(\mathbf{x}) \equiv \sum_{\alpha}^{N} \Psi_\alpha(\mathbf{x}) Q_\alpha$$

$$\text{error extremization: } \mathrm{GWS}^N \equiv \int_{\Omega} \Psi_\beta(\mathbf{x}) \mathcal{L}(q^N) \mathrm{d}\tau \equiv 0$$

$$\text{discretization: } \Omega \Rightarrow \Omega^h = \cup_e \Omega_e$$

$$\text{FE construction: } q^N \Rightarrow q^h \equiv \cup_e \{N(\mathbf{x})\}^T \{Q\}_e$$

$$\text{FE implementation: } \mathrm{GWS}^h \Rightarrow \sum_e \{\mathrm{WS}\}_e = \{0\}$$

$$\text{linear algebra: } [\mathrm{Matrix}]\{Q\} = \{\mathrm{b}\}$$

$$\text{error estimation: solution-adapted } \Omega^h \text{ refinement}$$

References

[1] Oden, J.T. and Reddy, J.N. (1976) *An Introduction to the Mathematical Theory of Finite Elements*, Wiley, New York.
[2] Moore, S., Baker, A.J., Dongarra, J., Halloy, C. and Ng, C. (2002) Active Netlib: An Active mathematical software collection for inquiry-based computational science and engineering education. *J. Digit. Inf.*, **2**.
[3] www.mathworks.com/Matlab.
[4] www.comsol.com.
[5] Baker, A.J. (2013) *Optimal Modified Continuous Galerkin CFD*, Wiley, London.