Aerospace design: a complex task

Michael Giles
Oxford University Computing Laboratory

April 25th, 1997

Overview

- hierarchical product definition
- hierarchical design
- different design philosophies and methods
- sensitivity analysis

WARNING: I am <u>not</u> a designer, so do I really know what design is about? My aim is to provoke thought and discussion.

Sources of Complexity

- lots of components
- lots of design parameters
- lots of design constraints and requirements
- multidisciplinary

In aerospace this is aggravated by the huge computational cost of detailed analysis.

Hierarchical EPD

Handling geometric complexity requires hierarchical electronic product definition (EPD)

- at lowest level, very simple functional description of major components appropriate to preliminary design
- at higher levels, increasing amount of detail as needed, for example, by CFD and structural analysis packages

Hierarchical EPD

Example: aircraft

Level 1	aircraft weight, wingspan, cruising speed
Level 2	wing/fuselage geometry
Level 3	engines, tail, winglets
Level 4	high-lift flaps & slats, take-off climb rate
Level 5	control surfaces, fairings, desired roll rate

Hierarchical EPD

Example: turbine vane

Level 1	number of blades, hub/tip radius, throat area, mass flow, inflow/outflow angles
Level 2	camber/thickness distribution, cooling mass flow
Level 3	fillets at hub and tip junctions
Level 4	film cooling holes and slots, temperature of coolant supply
Level 5	alloy type and thermal properties

Hierarchical EPD

- parameters at each level of EPD hierarchy form collective design parameters
- altering any parameter, at any level, defines a parametric change which is inherited at higher levels of the EPD system
- grid generator must be able to respond to this to define perturbed grids
- parametric solids-based core to CAD system may be essential to provide this capability

Hierarchical Design

Handling the very large number of design parameters in an aerospace system also requires a hierarchical approach

- to limit the number of 'active' design parameters
- to minimise the computational cost

Each level of design works with the appropriate level of EPD definition.

Hierarchical Design

Preliminary design

- uses very simple modelling of the overall system, with a lot of empiricism from past designs
- considers important trade-offs between different components
- often involves integer design parameters
- often aims to minimise airlines' operating costs, or maximise manufacturers' profit

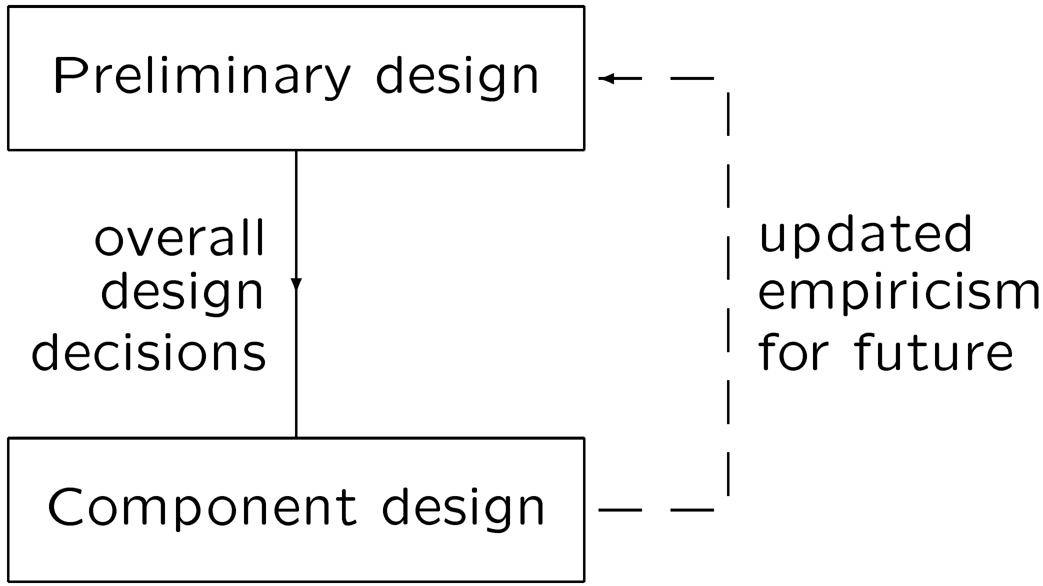
Hierarchical Design

Component design

- uses very detailed mathematical modelling with few approximations
- has to satisfy functional requirements and lots of constraints imposed by preliminary design
- often aims to minimise drag/loss, but maybe not directly

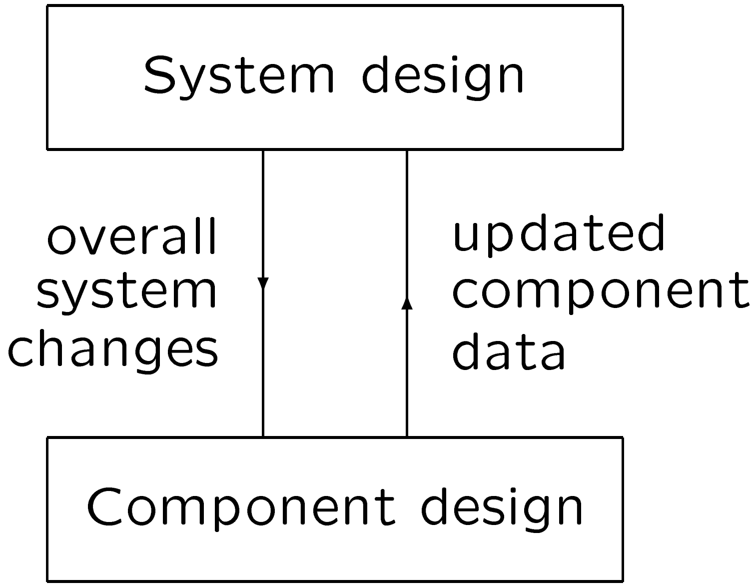
Hierarchical Design

Current system: preliminary design followed by component design



Hierarchical Design

Future: tightly-coupled two-level design



Requires a well-integrated design system, much of which is computer driven.

Design philosophies

Design parameters:	$\boldsymbol{\alpha}$
CFD/structural variables:	$\boldsymbol{U}$
Discrete equations:	$\boldsymbol{F}(\boldsymbol{U}, \boldsymbol{\alpha}) = 0$
Equality constraints:	$\boldsymbol{E}(\boldsymbol{U}, \boldsymbol{\alpha}) = 0$
Inequality constraints:	$\boldsymbol{C}(\boldsymbol{U}, \boldsymbol{\alpha}) \geq 0$

Design philosophies

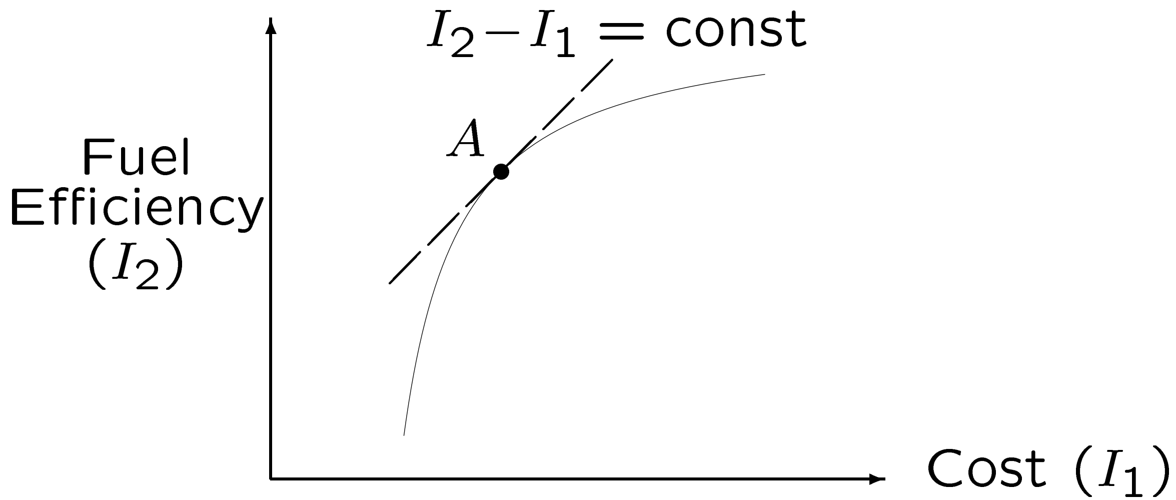
1) Define an objective function $I(\boldsymbol{U}, \boldsymbol{\alpha})$ and leave it to a black-box optimiser

- may be appropriate for preliminary design
- designer responsible for defining design space, constraints and objective function, and monitoring design evolution

Design philosophies

2) Define more than one objective function and view trade-off curve before choosing optimum

- puts the designer more in the loop
- may be appropriate for multidisciplinary applications



Design philosophies

3) Design system computes sensitivities with respect to design parameters but designer specifies design changes

- puts the designer totally in charge
- allows the designer to keep in mind other constraints not easily quantified
- design system could aid designer by ensuring some constraints are automatically satisfied

Design methods

Global optimisation in preliminary design: genetic algorithms and other stochastic optimisation methods

- good at finding global optimum not just a local optimum
- well suited to optimisation involving integer parameters
- computational cost acceptable because of low cost of empirical modelling

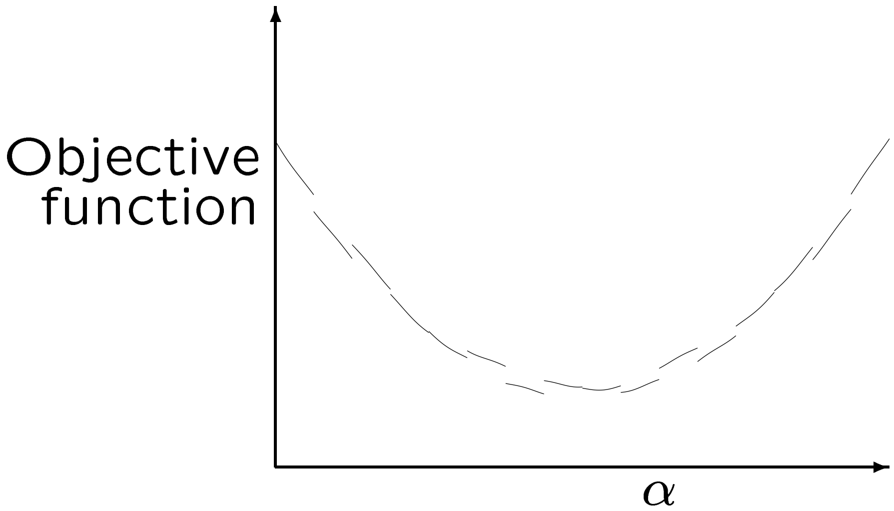
Design methods

Local optimisation in system and component design: gradient-based methods using (approximate) sensitivities

- lots of methods for unconstrained and constrained optimisation
- objective functions will not be smooth (discontinuities and little ‘ripples’) which could cause problems
- computationally less costly than stochastic optimisation (I think)

Design methods

- discontinuities due to changing grid topology
- ripples due to changing shock position relative to grid



Design methods

One approach to coping with these difficulties is to construct a least-squares approximation by a smooth function, sometimes called a ‘response surface’

Optimising this smooth approximation subject to smoothed constraints would be a relatively easy task.

Sensitivity Calculations

Nonlinear:

- perturb each component of $\boldsymbol{\alpha}$ in turn, compute new solutions and use to get approximate gradient
- easy to implement, trivial to parallelise
- expensive for large numbers of design parameters

Sensitivity Calculations

Linear:

- solve linearised discrete equations, like nonlinear treatment with very small perturbations
- no cost benefits compared to nonlinear treatment in most cases
- additional effort of writing linear code

Sensitivity Calculations

One advantage of direct linear/nonlinear sensitivity approach is Quasi-Newton optimisation for least-squares applications

Suppose we wish to minimise

$$I(\boldsymbol{\alpha}) = \sum_n \left(p(\boldsymbol{x}_n, \boldsymbol{\alpha}) - p_{des}(\boldsymbol{x}_n)\right)^2 \Delta s$$

At a minimum, we require

$$\frac{\partial I}{\partial \alpha_i} = 2 \sum_n \frac{\partial p}{\partial \alpha_i} \left(p(\boldsymbol{x}_n, \boldsymbol{\alpha}) - p_{des}(\boldsymbol{x}_n)\right) \Delta s = 0$$

Sensitivity Calculations

Solving this set of simultaneous equations using Newton-Raphson gives

$$\boldsymbol{A}(\boldsymbol{\alpha}^{n+1}-\boldsymbol{\alpha}^{n}) = -\boldsymbol{r}^{n}$$

where

$$r_i^n = \sum_n \frac{\partial p}{\partial \alpha_i} \left(p(\boldsymbol{x}_n, \boldsymbol{\alpha}) - p_{des}(\boldsymbol{x}_n)\right) \Delta s$$

and

$$A_{ij} = \sum_n \left(\frac{\partial p}{\partial \alpha_i}\frac{\partial p}{\partial \alpha_j} + \frac{\partial^2 p}{\partial \alpha_i \partial \alpha_j}(p - p_{des}) \right) \Delta s$$

Sensitivity Calculations

Neglecting the second-derivative term gives the Quasi-Newton method which converges quickly to the minimum if $p - p_{des}$ is small.

This can also be viewed as minimising the quadratic approximation

$$I \approx \sum_n \left(p(\boldsymbol{x}_n, \boldsymbol{\alpha}^n) + \frac{\partial p}{\partial \boldsymbol{\alpha}} \tilde{\boldsymbol{\alpha}} - p_{des}(\boldsymbol{x}_n) \right)^2 \Delta s$$

Sensitivity Calculations

Adjoint method:

- based on linear approach
- reduces number of calculations to one for each objective function and constraint function
- cost of each optimisation step is independent of number of design parameters, so can have lots; however number of steps may increase with number of parameters

Sensitivity Calculations

Sensitivity information can show significance of constraints imposed in preliminary design – feedback is crucial for better preliminary design trade-offs

Also useful in other areas:

- manufacturing tolerances
- risk management
- strategic research planning

What Would I Recommend?

A hierarchical solution:

- genetic algorithms for black-box optimisation of preliminary design
- for component design, start with few design variables, direct sensitivity analysis and optimisation by the designer (using response surface if necessary)
- for final refinement, add additional design variables and switch to adjoint-based optimisation