Part II – Decision Systems for Engineering Design
Lecture 4: Multi-objective Optimization II

Robin Purshouse
University of Sheffield

Spring Semester 2023/24

1 Lecture overview

This lecture continues the introduction to multi-objective optimization as a core element of a decision system for engineering design.

In this lecture, we will cover:

- Decomposition-based algorithms;
- Set-based algorithms.

During the lecture we will continue to work with the unconstrained multi-objective optimization problem (MOP) from the previous lecture:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \mathbf{z} = \mathbf{f}(\mathbf{x}, \mathbf{p}) \\ & \mathbf{x} \in \mathcal{D} \\ & \mathbf{p} \leftarrow \mathcal{P} \end{aligned} \tag{1}$$

where $\mathbf{x}$ is a candidate design from a design space $\mathcal{D}$, $\mathbf{z}$ is a vector of performance criteria, and $\mathbf{f}(.)$ is a vector of evaluation functions that estimate how a design will perform against the criteria when using a nominal parameterisation $\mathbf{p} = \mathbf{p}_0$ from parameter space $\mathcal{P}$.

2 Decomposition-based decision systems

2.1 Introduction to decomposition

Decomposition-based methods transform the vector of performance criteria $\mathbf{z}$ into a scalar value ζ. This is achieved using a *scalarising function* $\zeta = s(\mathbf{z}, \mathbf{w}, \mathbf{z}^{\text{ref}})$ where $\mathbf{w}$ is a vector of weighting parameters and $\mathbf{z}^{\text{ref}}$ is a reference point in performance space.

The scalarising function $s(.)$ is chosen in such a way that optimization of the function will yield a Pareto optimal design from the efficient set $\mathbf{X}^\star$ (see Equation 4 from Lecture 3). For the purposes of *a posteriori* decision making, the idea is that, by changing $\mathbf{w}$ and/or $\mathbf{z}^\star$ and then re-optimizing $s(.)$, designs from different locations on the Pareto front can be found.

2.2 Scalarising functions

In this section, we consider the most popular forms of scalarising function used in decomposition-based approaches.

2.2.1 Weighted sum

The *weighted sum* scalarising function is given by:

$$\zeta = s(\mathbf{x}|\mathbf{w}) = \sum_{i=1}^{M} w_i z_i(\mathbf{x}) \tag{2}$$

where $w_i \geq 0 \; \forall i$, $\sum_{i=1}^{M} w_i = 1$, and M is the number of objectives.

An example of the weighted sum scalarising function for a given $\mathbf{w}$ is shown in Figure 1. For convex Pareto fronts all points on the front can be found, given a suitable choice of $\mathbf{w}$. However, concave regions of Pareto fronts **cannot be found** using this approach. The reason is that the boundaries of the concave region will always produce a better value for ζ than the interior of the concave region *for any choice of* $\mathbf{w}$. We must therefore be careful to ensure that we are dealing with a convex Pareto front prior to application of any weighted sum decomposition method (fortunately there is some evidence that many real-world problems are characterised by convex fronts).

2.2.2 Reference point

The *reference point* or *achievement* scalarising function is given by:

$$\zeta = s(\mathbf{x}|\mathbf{z}^{\text{ref}}) = \sum_{i=1}^{M} (z_i(\mathbf{x}) - z_i^{\text{ref}}) \tag{3}$$

where z_i^{ref} is the ith component of a reference point in performance space.

If $\mathbf{z}^{\text{ref}}$ relates to a level of achievement that is of interest to the decision maker then optimization of the reference point scalarisation function will produce the closest

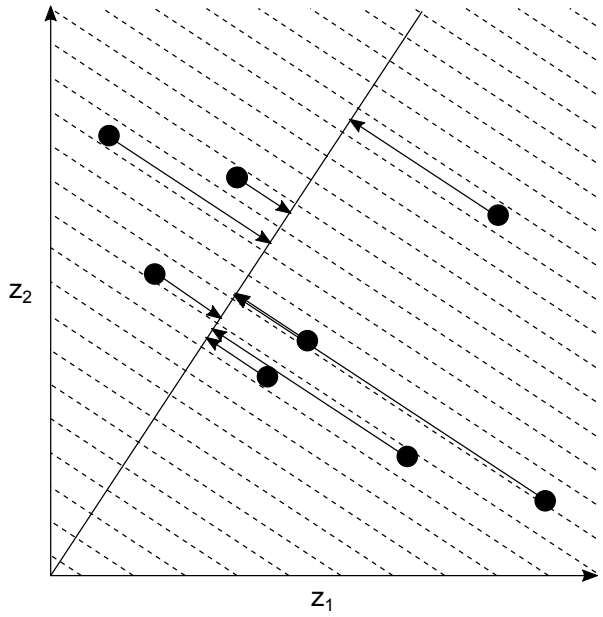

Figure 1: Weighted sum scalarisation of eight performance vectors for an arbitrary weight vector $\mathbf{w_1}$. The arrows indicate the mapping to the line $\zeta(\mathbf{x}|\mathbf{w} = \mathbf{w_1})$. The dashed lines indicate contours in ζ.

Pareto optimal point to that level of achievement. Further, all points on the Pareto front can be found by specifying alternative choices for $\mathbf{z}^{\text{ref}}$.

2.2.3 Weighted metric

The *weighted metric* combines and generalises the weighted sum and reference point approaches. The scalarising function is given by:

$$\zeta = s(\mathbf{x}|\mathbf{w}, \mathbf{z}^{\text{ref}}, l) = \left(\sum_{i=1}^{M} w_i (z_i(\mathbf{x}) - z_i^{\text{ref}})^l \right)^{1/l} \tag{4}$$

where $1 \leq l \leq \infty$ is the choice of norm and other variables are as previously introduced.

For suitably varying $\mathbf{w}$ and/or $\mathbf{z}^{\text{ref}}$, optimization of the corresponding weighted metric scalarising function will yield the full range of solutions on the Pareto front. However note that with $l = 1$ and $\mathbf{z}^{\text{ref}}$ fixed, the function is equivalent to the weighted sum and will not be able to identify concave areas of the front.

2.2.4 Tchebycheff metric

A popular configuration of the weighted metric scalarisation function given in Equation 4 is to set $l = \infty$ and $\mathbf{z}^{\text{ref}}$ to the *ideal* point of best possible performance in every criterion: $\mathbf{z}^{\text{ref}} = \mathbf{z}^{\star}$. The resulting function is known as the *Tchebycheff* (or *Chebyshev*) scalarising function and can be equivalently formulated as:

$$\zeta = s(\mathbf{x}|\mathbf{w}, \mathbf{z}^{\text{ref}} = \mathbf{z}^{\star}, l = \infty) = \max_i \left[w_i (z_i(\mathbf{x}) - z_i^{\star}) \right] \tag{5}$$

An example of the Tchebycheff scalarising function for a given $\mathbf{w}$ is shown in Figure 2.

Minimisation of Equation 5 will yield at least a weakly Pareto optimal solution. As usual, by varying $\mathbf{w}$, multiple such solutions can be found. To ensure that the search progresses beyond weakly optimal solutions to the Pareto optimal solutions (i.e. down or across the lines of equal ζ according to the $(l = \infty)$-norm), Equation 5 can be extended to form the *augmented Tchebycheff* scalarising function:

$$\zeta = s(\mathbf{x}|\mathbf{w}, \mathbf{z}^{\text{ref}} = \mathbf{z}^{\star}, l = \infty, \rho) = \max_i \left[w_i (z_i(\mathbf{x}) - z_i^{\star}) \right] + \rho \sum_{i=1}^{M} (z_i(\mathbf{x}) - z_i^{\star}) \tag{6}$$

where ρ is a sufficiently small positive scalar constant.

2.3 Normalisation

Decomposition-based approaches require objectives to be aggregated together. Such aggregation can be problematic in cases where the different performance criteria use

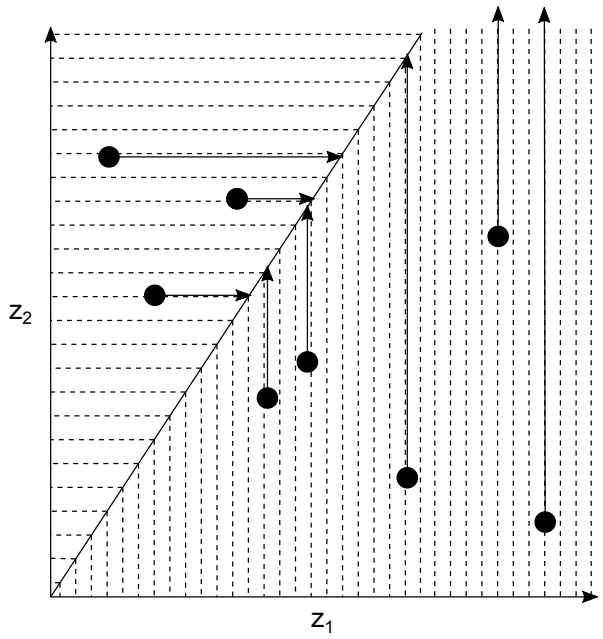

Figure 2: Tchebycheff scalarisation of eight performance vectors for the given weight vector $\mathbf{w_1}$. The arrows indicate the mapping to the line $\zeta(\mathbf{x}|\mathbf{w} = \mathbf{w_1}, \mathbf{z}^{\text{ref}} = \mathbf{0}, l = \infty)$. The dashed lines indicate contours in ζ.

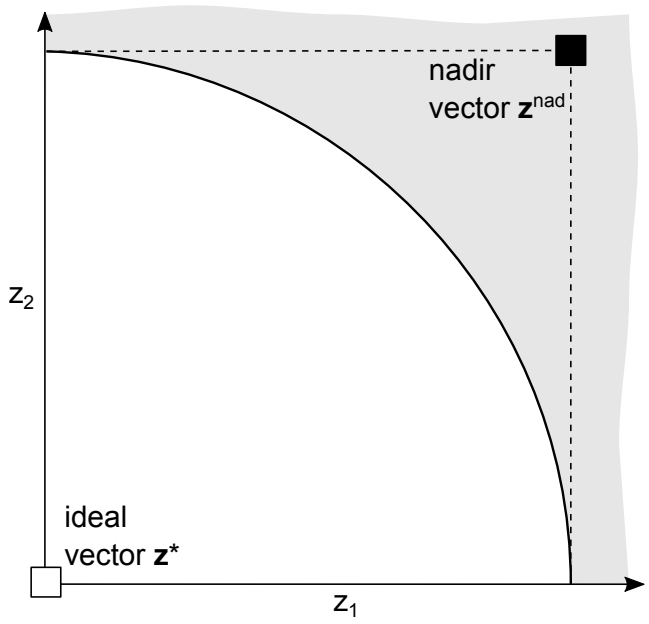

Figure 3: Ideal and nadir vectors of performance, with respect to the Pareto front

different units (e.g. grams of CO_2 for a fuel economy criterion; kilometres per hour for a maximum speed criterion) or are of different orders of magnitude.

To avoid problems, it is recommended to normalise the objectives prior to scalarisation. Normalisation is usually performed according to the following equation:

$$\bar{z}_i(\mathbf{x}) = \frac{z_i(\mathbf{x}) - z_i^\star}{z_i^{\text{nad}} - z_i^\star} \tag{7}$$

where $z_i^\star$ is the ith component of the ideal vector (introduced previously) and z_i^{nad} is the ith component of the *nadir* vector of worst performance across all Pareto optimal solutions. Figure 3 illustrates ideal and nadir vectors for a two-dimensional performance space. Note that estimating the ideal and nadir vectors in advance of optimization can be problematic (particularly the estimation of nadir vectors)—this is a key limitation of the decomposition-based approach.

w_1	w_2	w_3
0	0	1
0	0.333	0.667
0	0.667	0.333
0	1	0
0.333	0	0.667
0.333	0.333	0.333
0.333	0.667	0
0.667	0	0.333
0.667	0.333	0
1	0	0

Table 1: An example weight vector set $\mathcal{W}$ for a three-objective problem with three divisions per objective

2.4 MOEA/D

The *multi-objective evolutionary algorithm based on decomposition (MOEA/D)* is the most well-known optimizer based on the decomposition concept. This section of the lecture provides an introduction to the key concepts that underpin the algorithm.

2.4.1 Scalarising functions as sub-problems

To use MOEA/D within a decision system it is first necessary to define a scalarisation function and associated parameters. The algorithm can work with any of the scalarising functions introduced above (as well as others) but is usually configured to use the Tchebycheff metric. The scalarising function is then used to define a number of sub-problems which will be optimized simultaneously, where each sub-problem corresponds to a particular choice of weight vectors $\mathbf{w}$.

The set of weight vectors should aim to capture the overall characteristics of the Pareto front, typically aiming for an even coverage. In the basic MOEA/D algorithm, weight vectors are constructed according to an $\{M, h\}$-simplex-lattice design:

$$\mathcal{W} = \left\{ \mathbf{w} = [w_1, \ldots, w_M] \mid \sum_{i=1}^{M} w_i = 1 \wedge w_i \in \left\{ \frac{0}{h}, \frac{1}{h}, \ldots, \frac{h}{h} \right\} \forall i \right\} \tag{8}$$

where h defines the number of divisions for each objective. Under the simplex-lattice design, the set of weight vectors constitutes all possible combinations of weights that satisfy Equation 8. The total number of weight vectors produced will be $N = {}^{h+M-1}C_{M-1}$. For example, a $\{3, 3\}$-simplex-lattice creates the set of ${}^{5}C_2 = 10$ weight vectors shown in Table 1.

Note that the simplex-lattice design produces vectors that are evenly distributed in the space of weights; but this will only produce an even distribution on the Pareto front if the latter is linear and has been normalised correctly. Therefore for Pareto fronts that are convex, concave, or of mixed geometry, or cases where normalisation is not perfect, it may be better to use a different approach. This is an advanced topic that lies outside the scope of the module.

2.4.2 Sub-problem neighbourhoods

Once the sub-problems have been defined, a neighbourhood $B(j)$ around each sub-problem j is established. The neighbourhoods are defined by calculating the Euclidean distances between all pairs of weight vectors and then, for every sub-problem j, choosing the sub-problems with the closest T weight vectors as the neighbours of j. Here the sub-problem itself is included in the count of T, so a neighbourhood of size $T = 2$ would include sub-problem j and one other sub-problem that is closest to j in weight space.

2.4.3 The algorithm

MOEA/D allocates a design to each sub-problem. It then uses the neighbourhoods around the sub-problems to perform localised selection-for-variation, variation, and selection-for-survival operations.

1. Initialise the iteration counter $t = 0$.
2. Generate a candidate design $\mathbf{x}_j^t$ for every sub-problem, i.e. $j = 1 \dots N$.
3. Estimate the ideal vector $\hat{\mathbf{z}}^\star$ using the initial population or prior knowledge.
4. For every sub-problem $j = 1 \dots N$:
   (a) Randomly select the current designs from two sub-problems in the neighbourhood $B(j)$ and apply variation operators to these designs to generate a new candidate design $\mathbf{y}_j^t$.
   (b) Evaluate new design $\mathbf{y}_j^t$.
   (c) Re-estimate the ideal vector: $\forall i \in 1, \dots, M$, if $z_i(\mathbf{y}_j^t) < \hat{z}_i^\star$ then $\hat{z}_i^\star = z_i(\mathbf{y}_j^t)$
   (d) Across the neighbourhood of j, replace current design $\mathbf{x}_k^t$ with design $\mathbf{y}_j^t$ if $s(\mathbf{z}(\mathbf{y}_j^t), \mathbf{w}_k, \hat{\mathbf{z}}^\star) \leq s(\mathbf{z}(\mathbf{x}_k^t), \mathbf{w}_k, \hat{\mathbf{z}}^\star)$ for neighbour k.
5. Increment the iteration counter $t = t + 1$.
6. If a convergence criterion is satisfied then stop; otherwise go to Step 4.

MOEA/D has three principal benefits as an algorithm: (i) it is parallelisable; (ii) it allows specific directions to be targeted (assuming weight vectors can be defined appropriately); (iii) the localisation of its operators helps improve convergence efficiency by only sharing information across similar areas of performance space. This latter feature helps avoid the phenomenon of *lethals*, where recombination operations on very different parent solutions tend to produce poor performing child solutions.

MOEA/D's principal drawback relates to the need to perform effective normalisation. The performance of the algorithm may also be sensitive to the choice of its neighbourhood size parameter, T.

3 Set-based decision systems

3.1 Background

So far we have considered the Pareto-based and decomposition-based classes of multi-objective optimizers. Both these classes are population-based, but are concerned with the performance of individual candidate designs in comparison to others. It is hoped that good individual levels of performance, when taken across a population, will yield a good overall approximation set for the Pareto front. The final class of approach we will consider are *set-based* methods, which explicitly consider the performance of a population (or set) of solutions as a whole.

Unusually, set-based methods arose from performance testing of existing multi-objective optimizers. Performance indicators were developed that summarised the approximation set that was generated by one optimizer and allowed it to be compared to the approximation set generated by a different optimizer. Researchers quickly realised that these performance measures could actually be used as fitness measures to drive the convergence of an algorithm. For this reason, set-based methods are also sometimes described as *indicator-based* methods.

A number of different performance indicators have been harnessed for use in set-based optimization. In this module, we will focus on one particular indicator known as *hypervolume* or the $\mathcal{S}$*-metric*.

3.2 Hypervolume

Given an approximation set, $\mathcal{A}$, the hypervolume measures the volume of the performance space that is weakly dominated by the approximation set's members, where the volume is bounded by some dominated reference point, $\mathbf{z}^{\text{ref}}$. Formally, the hypervolume, $S(\mathcal{A}, \mathbf{z}^{\text{ref}})$, is defined as:

$$S(\mathcal{A}, \mathbf{z}^{\text{ref}}) = \mu\left(\bigcup_{\mathbf{x}\in\mathcal{A}} \left\{\mathbf{z}' : \mathbf{z}(\mathbf{x}) \preceq \mathbf{z}' \preceq \mathbf{z}^{\text{ref}}\right\}\right) \tag{9}$$

where $\mu(.)$ is the Lebesgue measure of the set (or, in more simple terms, the volume defined by the set).

As an indicator, hypervolume has two very desirable properties: (i) if $S(\mathcal{A}_1, \mathbf{z}^{\text{ref}}) > S(\mathcal{A}_2, \mathbf{z}^{\text{ref}})$ then approximation set $\mathcal{A}_1$ cannot be dominated by approximation set $\mathcal{A}_2$ (i.e. it cannot be worse in terms of Pareto dominance); and (ii) hypervolume is maximised, for a given $\mathbf{z}^{\text{ref}}$, if and only if $\mathcal{A}$ is the set of all Pareto optimal designs that map to unique points in performance space. An example calculation of hypervolume for a performance space with two objectives is shown in Figure 4.

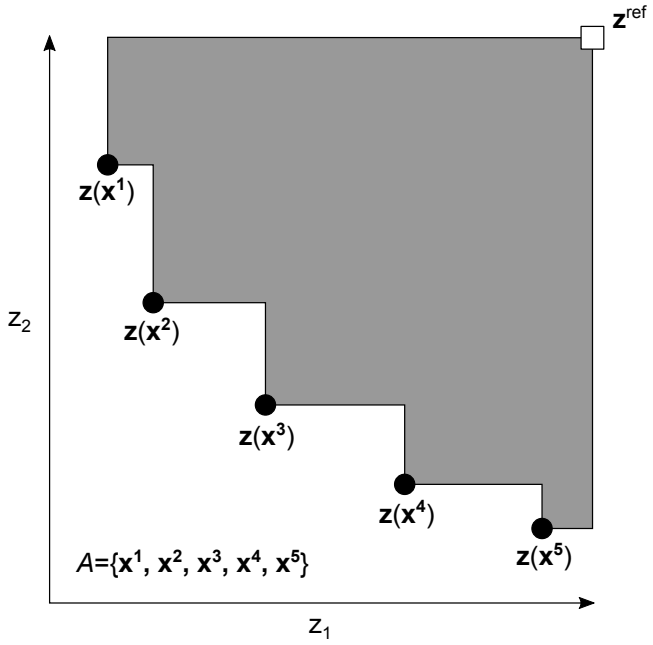

Figure 4: Illustration of the hypervolume indicator $S(\mathcal{A}, \mathbf{z}^{\text{ref}})$ for an approximation set with five members. The hypervolume is shown by the grey area.

3.3 SMS-EMOA

The *$\mathcal{S}$-metric selection evolutionary multi-objective optimization algorithm (SMS-EMOA)* is one of the most well-known set-based optimizers. The algorithm attempts to find a set of Pareto optimal designs that maximises the hypervolume indicator, subject to a defined finite approximation set size.

Like NSGA-II, SMS-EMOA uses non-dominated sorting as the primary component of its selection mechanism. However, the crowding distance used in NSGA-II is replaced with a measure based on hypervolume.

3.3.1 The algorithm

SMS-EMOA proceeds as follows:

1. Initialise the iteration counter $t = 0$.
2. Generate a population $\mathcal{P}[0]$ of N candidate designs.
3. Randomly select two designs from the population $\mathcal{P}[t]$ and apply variation operators to these designs to generate a new candidate design $\mathbf{y}^t$.
4. Evaluate new design $\mathbf{y}^t$.
5. Apply the `Reduce` procedure to generate the new population $\mathcal{P}[t+1]$ from $\{\mathcal{P}[t] \cup \mathbf{y}^t\}$.
6. Increment the iteration counter $t = t + 1$.
7. If a convergence criterion is satisfied then stop; otherwise go to Step 3.

3.3.2 The `Reduce` procedure

The `Reduce` procedure operates in a similar fashion to NSGA-II's selection-for-survival operator. The new population is defined as the null set, $\mathcal{P}[t+1] = \varnothing$, and is incrementally filled using ranks obtained from the non-dominated sorting procedure (see Section 4.2.1 in Lecture 3). When a rank is reached that would lead to an overspill of the population, the designs in the final rank are inspected for the *contributions* that each makes to the overall hypervolume associated with that rank. The design with the lowest hypervolume contribution is then discarded. An example of this process is shown in Figure 5.

An advantage of the SMS-EMOA algorithm is that the ranking of solutions according to hypervolume contribution is invariant to scaling of the objectives, making normalisation less of a concern in comparison to MOEA/D. However the hypervolume calculation is sensitive to choice of the reference point $\mathbf{z}^{\text{ref}}$. The hypervolume is also computationally expensive to calculate analytically, although it can be approximated more cheaply using Monte Carlo approaches.

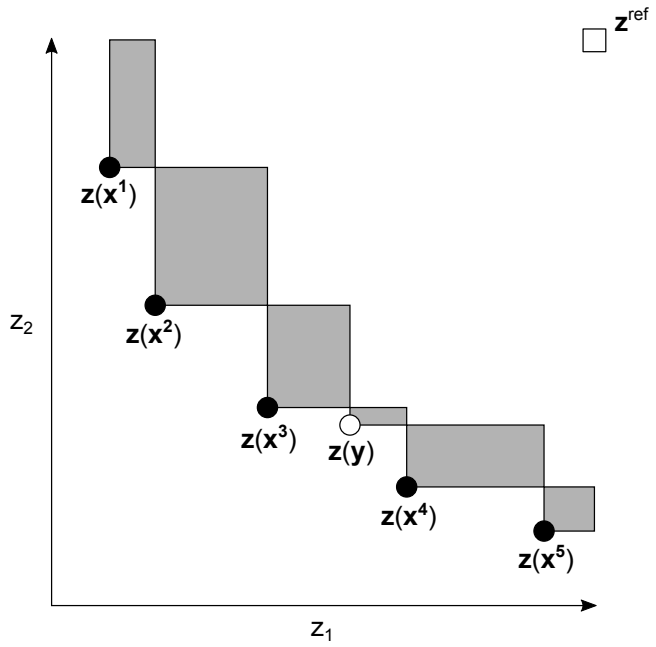

Figure 5: Hypervolume contributions: a new candidate design $\mathbf{y}$ is considered for inclusion in the new population of five solutions. The hypervolume contributions of each design are shown by the grey areas. The new design would not be retained by SMS-EMOA's `Reduce` algorithm.

4 Further reading

- Zhang Q., Li H. MOEA/D: A multi-objective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation* 2007;11:712–731.
- Beume N., Naujoks B., Emmerich M., SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research* 2007;181:1653–1669.