Noname manuscript No.
(will be inserted by the editor)

Thrust vs. Endurance Trade-off Optimization in Swimming

Christopher WG Phillips · Dominic A Hudson · Stephen R Turnock · Alexander IJ Forrester

Received: date / Accepted: date

Abstract In this paper we apply methods typically used in engineering applications to the optimization of human locomotion, more specifically a swimmer's underwater 'dolphin kick'. This is a dual-objective problem where we search for the optimal trade-off between thrust (simulated using Lighthill's fish propulsion method) and the force in the muscles to produce this thrust (simulated using musculoskeletal modelling). The expense of the analyses leads us to use a surrogate modelling based optimization technique (multi-objective expected improvement using Kriging). Our results indicate that optimal human motion does, in many respects follow that of fish, with low frequency eel-like techniques suitable for endurance and a high frequency tuna-like kick for high thrust. *Matlab*® code, including thrust and muscle activity models is included as supplementary material.

Keywords multiobjective · surrogate modelling · Kriging · expected improvement · dolphin kick · musculoskeletal modelling · swimming

1 Introduction

Humans propel themselves fastest underwater by mimicking aquatic life; using the so-called dolphin kick[1]. Due to the increased speed possible with this technique, swimmers are limited to 15 metres of underwater swimming per length (in official races). To maximise the effectiveness of these underwater periods, we wish to understand the trade-off between speed and endurance, and find a dolphin kick technique that has the correct balance for a given race length.

Alexander IJ Forrester
University of Southampton, UK
Tel.: +44 (0)23 8059 2713
E-mail: Alexander.Forrester@soton.ac.uk

[1] Other terms for this technique are fly-kick or human underwater undulatory swimming.

To continue the analogy with aquatic life, consider the difference between an eel and a tuna. The former swims entire oceans at minimal effort while the latter is optimized for speed. The technique is very different: the whole-body *anguilliform* of the eel and the tail-dominated *tuniform* of the tuna. A halfway house would be, for example, the *carangiform* of the mackerel. Naturally humans are limited in their ability to mimic these forms precisely, due to fewer joints in their 'tails' and non-symmetric flexion/extension in the sagittal plane, but we anticipate that similar trends in technique and performance will hold true. Through fluid dynamic and musculoskeletal simulation, coupled with a surrogate modelling based multi-objective optimization strategy, we aim to obtain an optimal trade-off of speed versus endurance for human swimming. In a similar vein as cost/performance tradeoffs drive decisions in engineering design, provision of services, etc., here identifying the endurance (cost) vs. thrust (performance) tradeoff could allow coaches and athletes to select optimal techniques for given race distances.

Our aim requires us to parameterise the dolphin kick technique; allowing us to vary a simulated kick-cycle to move between optimal strokes in a multi-dimensional hyperspace. The parameterisation must be parsimonious to reduce the size of this optimization search space, but flexible enough to produce a wide range of kick forms, which are also physically realisable. We discuss this parameterisation in Section 2. Despite our efforts at parsimony, the potential for multiple local optima and the expense of musculoskeletal simulation dictates that an efficient global optimization method is needed. We employ a multi-objective expected improvement based search using Guassian process (Kriging) models, which is presented in Section 3. The fluid dynamics and musculoskeletal models, used to simulate the performance of the different parameterised techniques chosen by the optimiser, are presented in Sections 4 and 5, before results are presented and discussed in section 6.

2 Dolphin kick parameterisation

As a basis for our parameterisation of the dolphin kick, we analyse a video of a male world record holding swimmer. The video is obtained using an underwater camera, translating parallel to the swimmer (with the swimmer appearing essentially stationary in x-direction in the video). The x velocity (required later for Equation 13) is measured using a trailing line attached to the swimmer, which is unwound from a rotary encoder as the swimmer propels themselves forward (Phillips et al., 2014) – as depicted in in Figure 1. The line is attached at the swimmer's side at the centre of rotation of their pelvis segment. In doing so we minimise the impact on the swimmer's motion and reduce artificial surges in velocity caused by local hip rotation rather than true forward speed. The velocity data is collected at 250 Hz.

We assume a purely two-dimensional kick, i.e. both legs kick in unison, and capture the kinematics by recording the pixel location of anatomical landmarks at 25 frames per second for 10 kick-cycles. A sample frame, with these landmarks identified is shown in Figure 2. The process is completed using a *Matlab*® interface, via which landmarks are visually identified and 'clicked' upon, frame-by-frame. [2] Errors in this manual digitisation process will, naturally, impact on the fluid dynamic and musculoskeletal simulations. However, this experimental data collection is a means to obtain realistic baseline kinematics and small errors here will not propagate through to our final solution.

From Figure 2, it can be seen that the body has been simplified into seven segments; hand, arm, thorax & abdomen, pelvis, thigh, shank and foot (labelled 1 through 7 respectively). These segments are articulated by six joints: ankle, knee, hip, lumbosacral, shoulder and wrist. The body is orientated about the pelvis such that here there are two angles of interest: the pelvis angle with respect to the global frame of reference, and the pelvis-thorax angle. We therefore have a total of seven time-dependent angles to optimise over the kick-cycle. For the purposes of this study we will limit ourselves to the hip, pelvis-thorax and pelvis angles (angles between segments 5 & 4, 4 & 3 and the angle of segment 4, respectively). Experience shows that these are important parameters in coaching the technique and, being at the root of the connected and articulated system, will have greatest influence on resultant motion (Phillips, 2013).

The recorded kinematics are then parameterised with the mean and first two pairs of terms of a Fourier series fitted to each joint angle for the 10 kick-cycles. With three joints of interest this yields 15 variables for our two-objective trade-off optimization. The mean Fourier term of each angle is constrained to be between ±10° of the observed experimental mean. Similarly, the sine and cosine terms were constrained to be within ±25% of the experimental values.

A single cycle representation of the Fourier series fit of the ten kick-cycles of the experimental data is shown as the bold line in Figure 3. The raw data for the ten kick-cycles are also shown. The mean, minimum and maximum Pearson correlation coefficients between the Fourier series and these raw data are $\bar{\rho}_{\text{hip}} = 0.90^{1.00}_{0.78}$, $\bar{\rho}_{\text{pelvic pitch}} = 0.93^{0.99}_{0.84}$, & $\bar{\rho}_{\text{pelvis}-\text{thorax}} = 0.91^{0.99}_{0.77}$. These values close to one indicate the appropriateness of using a Fourier series to represent the kinematics. Optimizing the coefficients changes the amplitude and frequencies, but retains the characteristic rhythmic motion. The remaining joints are articulated using the mean and first two pairs of a Fourier series approximation of the original kinematics.

In the next section we discuss the method by which optimal values for these 15 Fourier terms is found (noting that we use the term optimal slightly loosely, as we are limited to the bounds shown in Figure 3 and, with a large (15-dimensional) search space, will never find truly optimal solutions in all but the most trivial problems).

[2] Automated contrast recognition software can be used for such a process, but here the reliability of manual digitisation outweighed the time-savings of automation.

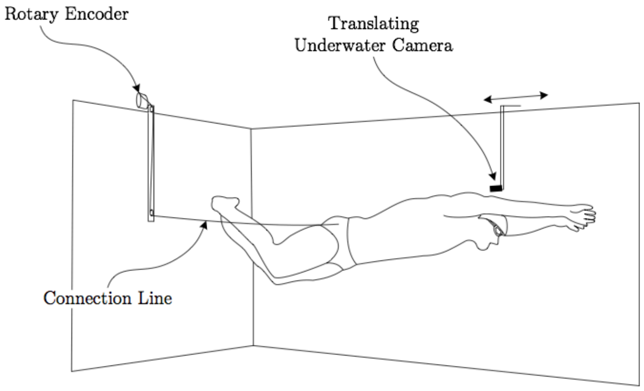

Fig. 1 A schematic of the video and velocity acquisition arrangement.

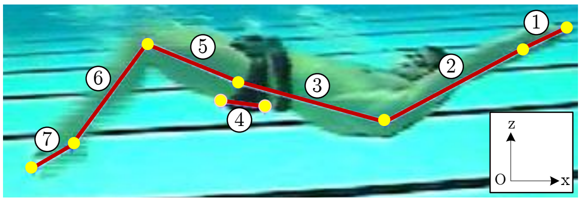

Fig. 2 Digitising the image to obtain the kinematics used in the thrust and musculoskeletal simulations. The hip, pelvis-thorax and pelvis rotations are defined by the angles between segments 5 & 4, 4 & 3 and the angle of segment 4, respectively)

3 A brief tutorial on the expected improvement approach to optimization

The thrust and endurance calculations discussed in Sections 5 and 4 provide the *objective functions* for our trade-off study. A single calculation of both objectives takes approximately 30 minutes on a desktop machine with 12 GB RAM and 2.67 GHz CPU. To keep within a one week time limit, we are therefore limited to 100s rather than 1000s of calculations. This effectively precludes the direct use of global search methods such as genetic algorithms (GAs). We therefore build surrogates of the objective functions, which take fractions of a second to evaluate and can be used in lieu of the original calculations when applying a GA. There is a large literature on the use of surrogate models in engineering, with a review by Forrester and Keane (2009) and a more detailed discussion of our chosen method, Kriging, by Jones, Schonlau, and Welch (1998). Rather than repeat these, and other works, in this section

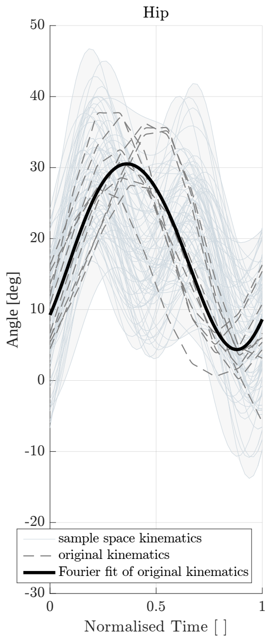

Fig. 3 The mean joint angle for one kick-cycle derived from experimental data is shown as a single bold line. Raw data for the ten kick-cycles are shown as dashed lines. The many thin, solid grey lines illustrate the joint angles from the initial optimal Latin hypercube sample of the search space.

we will take more of a tutorial approach, and have provided *Matlab*® code as supplementary material.[3]

3.1 Kriging

We start with a set of n experiments defined by n vectors of k design variables, $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(n)}$ ($k = 15$ for our dolphin kick parameterisation). There are many ways in which we could choose how many experiments to run and which combinations of design variables to use in each of these experiments. We will assume that the observed results will vary smoothly and continuously over our domain of interest and so, without any further knowledge of the domain, we are likely to achieve better models with experiments evenly distributed throughout the domain. To do this we maximise the minimum distance between any two experiments in the k-dimensional hypercube. This criterion would be satisfied by a grid of points, however, sampling only every corner of the hypercube

[3] A more extensive surrogate modelling toolbox for *Matlab*® is available at `www.optimization.codes` with an accompanying book (Forrester, Sóbester, and Keane, 2008). Another notable resource is the R package *DiceKriging* (Roustant, Ginsbourger, and Deville, 2012).

would require 2^k experiments, which is 32,768 in our case. This is clearly impracticable. We also wish to incorporate some randomness in the location of points to avoid potential problems with aliasing. This is when we sample at or above the frequency of a harmonic effect in the response (which we may expect here, given the Fourier series parameterisation) and so miss this effect. A maximin Latin hypercube design (Morris and Mitchell, 1995) can guard against this, whilst maintaining an even distribution of experiments, and is implemented in Listing 1 using our *Matlab*® toolbox.

In all but very low dimensional problems, the number of experiments is usually limited by financial and/or time constraints. Rules of thumb are to use $10k$ experiments (Loeppky, Sacks, and Welch, 2009) or, when going on to search and update the model with an expected improvement infill strategy, as we do here in the following sections, choose 1/3 of the total budget for the initial sample (Sóbester, Leary, and Keane, 2005). Here we use $n = 75$ and add 150 further points, from a total budget of 225.

Listing 1 *Matlab*® script to create an $n = 75$, $k = 2$ maximin Latin hypercube sample.

```
n=75; % no. points in initial sample
k=15; % no. variables
% call bestlh (last two arguments control maximin search)
ObjectiveInfo{1}.X=bestlh(n,k,75,10);
% same sample for both objectives
ObjectiveInfo{2}.X=ObjectiveInfo{1}.X;
```

These experiments produce $2n$ responses, $\mathbf{y}_j = \{y_j^{(1)}, y_j^{(2)}, ..., y_j^{(n)}\}^T$ for $j \in [1, m]$ where m is the number of objectives (here $m = 2$), which are calculated in Listing 2.

Listing 2 *Matlab*® script to calculate both objective functions at the sample locations.

```
for i=1:n
    ObjectiveInfo{1}.y(i,1)=thrust(ModelInfo.X(i,:));
    ObjectiveInfo{1}.y(i,1)=maxmuscleactivity(ModelInfo.X(i,:));
end
```

Kriging assumes that our observed data (i.e. here our thrust and endurance calculations) are random variables $Y(\mathbf{x}^{(1)}), Y(\mathbf{x}^{(2)}), ..., Y(\mathbf{x}^{(n)})$.[4] Building a Kriging model is a process of finding the parameters that describe the correlations between these random variables. We specify the form of the correlation as

$$\mathrm{cor}[Y(\mathbf{x}^{(i)}), Y(\mathbf{x}^{(l)})] = \exp\left[-\sum_{j=1}^{k} \theta_j \left(x_j^{(i)} - x_j^{(l)}\right)^2\right]. \tag{1}$$

This is a Gaussian correlation function[5], which has the intuitive property that as two experiments $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(l)}$ approach each other in the k dimen-

[4] In reality the thrust produce by and the endurance of a swimmer are clearly random variables – a slightly different result will be obtained every time they are measured –, but our computer simulations are not – they always return the same result. Nevertheless, Kriging can still provide remarkably good surrogates of such deterministic experiments.

[5] Other correlation functions are possible and may be more suitable for some problems – see, e.g.Ginsbourger, Durrande, and Roustant (2013).

sional hypercube, the correlation goes to unity, but as $|\mathbf{x}^{(i)} - \mathbf{x}^{(l)}| \to \infty$, $\mathrm{cor}[Y(\mathbf{x}^{(i)}), Y(\mathbf{x}^{(l)})] \to 0$. That is, two of the same experiments will have the same answer and two very different experiments will have different answers. The rate at which the correlation drops off is controlled by the θ_j parameter. If a variable has very little effect on the result of the experiment, $\theta_j \to 0$, while for a large effect $\theta_j >> 0$. Maximum likelihood estimates (MLEs) are found for the $\boldsymbol{\theta}$ parameters of each objective function, based on the observed data. To find these estimates we in fact minimize the negative of the *concentrated ln-likelihood function*:

$$\min_{\boldsymbol{\theta}} \frac{n}{2}\ln(\widehat{\sigma}^2) + \frac{1}{2}\ln|\boldsymbol{\Psi}|, \tag{2}$$

where $\widehat{\sigma}^2$ is a MLE of the variance in the data:

$$\widehat{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{1}\widehat{\mu})^T \boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\widehat{\mu})}{n}, \tag{3}$$

$\widehat{\mu}$ is a MLE of the mean of the data:

$$\widehat{\mu} = \frac{\mathbf{1}^T \boldsymbol{\Psi}^{-1}\mathbf{y}}{\mathbf{1}^T \boldsymbol{\Psi}^{-1}\mathbf{1}}, \tag{4}$$

$\boldsymbol{\Psi}$ is an $n \times n$ matrix of correlations between all data points (the correlations defined by equation 1):

$$\boldsymbol{\Psi} = \begin{pmatrix} \mathrm{cor}[Y(\mathbf{x}^{(1)}), Y(\mathbf{x}^{(1)})] & \dots & \mathrm{cor}[Y(\mathbf{x}^{(1)}), Y(\mathbf{x}^{(n)})] \\ \vdots & \ddots & \vdots \\ \mathrm{cor}[Y(\mathbf{x}^{(n)}), Y(\mathbf{x}^{(1)})] & \dots & \mathrm{cor}[Y(\mathbf{x}^{(n)}), Y(\mathbf{x}^{(n)})] \end{pmatrix}. \tag{5}$$

and $\mathbf{1}$ is an $n \times 1$ vector of ones. Listing 3 shows how this minimization can be performed in *Matlab*®. Lines 2 and 3 set upper and lower bounds for the $\boldsymbol{\theta}$ parameter before a for-loop cycles through the two objectives. Line 9 uses *Matlab*®'s genetic algorithm (GA) to minimize the output of this function. Line 12 calls `likelihood()` so that the correlation matrix and its Cholesky factorisation can be calculated based on the MLE for $\boldsymbol{\theta}$, and stored for use in future calculations.

Listing 3 *Matlab*® script to find a MLE for $\boldsymbol{\theta}$, build the correlation matrix, and find its Choloesky decomposition.

```
1  % set bounds on thetas
2  lowerTheta=ones(k,1).*-3;
3  upperTheta=ones(k,1).*2;
4  % set GA options as desired
5  options=gaoptimset('PopulationSize',10,'Generations',5);
6  % for each objective ...
7  for i=1:2
8      % run GA, calling likelihood()
9      ObjectiveInfo{i}.Theta=ga(@(x) likelihood(x,ObjectiveInfo{i})...
10     ,k,[],[],[],[],lowerTheta,upperTheta,[],options);
11     % and store matrices
12     [NegLnLike,ObjectiveInfo{i}.Psi,ObjectiveInfo{i}.U]=...
13     likelihood(ObjectiveInfo{i}.Theta,ObjectiveInfo{i});
14 end
```

With the model parameters estimated, predictions at new points can be made quickly using the Kriging mean:

$$\widehat{y}(\mathbf{x}^{(n+1)}) = \widehat{\mu} + \boldsymbol{\psi}^T \boldsymbol{\Psi}^{-1}(\mathbf{y} - \mathbf{1}\widehat{\mu}) \tag{6}$$

(where $\boldsymbol{\psi}$ is a vector of correlations of the point at which the prediction is to be made, $\mathbf{x}^{(n+1)}$, and the observed data), which is implemented in the function `predictor()`. For example, the script in Listing 4 will find values for both objectives at the centre of the search domain.

Listing 4 *Matlab*® script to find a MLE for $\boldsymbol{\theta}$, build the correlation matrix, and find its Choloesky decomposition.

```
ObjectiveInfo{1}.Option='Pred';
thrustPred=predictor(ones(1,15).*0.5,ObjectiveInfo{1});
ObjectiveInfo{2}.Option='Pred';
maxMusActPred=predictor(ones(1,15).*0.5,ObjectiveInfo{2});
```

3.2 Expected improvement

The quick-to-call models created in the previous section can now be used as surrogates for the original objective functions, and searched extensively by a range of optimisers. However, one must remember that these are just models and cannot necessarily be trusted to produce accurate results. Tentative optima can be validated by running the true codes at these locations in the search domain. This gives confidence in the values at these points, but cannot guarantee optimality. Indeed, it is unlikely that the region of the global optimum will be found by exploiting models built with such a sparse initial sample. There are a number of methods we can use to guard against possible errors in the Kriging models. Jones (2001) is an excellent taxonomy of these methods. The key benefit of Kriging (and other similar methods) is that, as well as the Kriging mean prediction in Equation 6, we have an estimate of the error in this prediction:

$$\widehat{s}^2(\mathbf{x}^{(n+1)}) = \widehat{\sigma}^2 \left[1 - \boldsymbol{\psi}^T \boldsymbol{\Psi}^{-1} \boldsymbol{\psi} + \frac{(1 - \mathbf{1}^T \boldsymbol{\Psi}^{-1} \boldsymbol{\psi})^2}{\mathbf{1}^T \boldsymbol{\Psi}^{-1} \mathbf{1}} \right]. \tag{7}$$

Sampling where this error is large will help to improve the model, rather than simply exploiting it, though more subtle criteria can be developed. Consider the situation in Figure 4; a rather deceptive one-dimensional function has been sampled at eight locations, which have failed to fall in the region of the global optimum where the Kriging prediction is a poor representation of the true function. A vertical Gaussian distribution has been plotted with its centre at the kriging mean and with variance from Equation 7. This distribution represents the values that the Kriging prediction could take. We naturally opt for the mean when making our prediction, but we can calculate the probability of other values. A useful quantity is the probability of improving on the best value so far. This integral is shown in bold in the figure. When searching

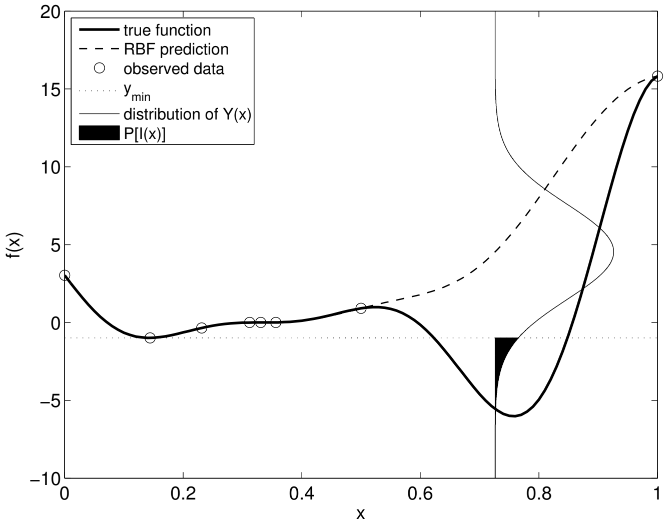

Fig. 4 A graphical representation of the expected improvement. The true function is $f(x) = (6x-2)^2 \sin(12x-4)$. This figure is reproduced from Forrester, Sóbester, and Keane (2008).

for useful improvements, a more pertinent criterion is the expectation of this improvement, which is calculated as:

$$E[I(\mathbf{x})] = \begin{cases} (y_{\min} - \widehat{y}(\mathbf{x}))\Phi\left(\frac{(y_{\min}-\widehat{y}(\mathbf{x}))}{\widehat{s}(\mathbf{x})}\right) + \widehat{s}(\mathbf{x})\phi\left(\frac{(y_{\min}-\widehat{y}(\mathbf{x}))}{\widehat{s}(\mathbf{x})}\right) & \text{if } \widehat{s}(\mathbf{x}) > 0 \\ 0 & \text{if } \widehat{s}(\mathbf{x}) = 0 \end{cases}, \tag{8}$$

where $\Phi(.)$ and $\phi(.)$ are the normal cumulative density function (cdf) and probability density function (pdf) respectively. Note that $\mathbf{x}^{(n+1)}$ has been abreviated to $\mathbf{x}$ for clarity.

Equation 8 may implemented using the code in Listing 4, also using the function `predictor()`, but with `ObjectiveInfo{.}.Option='NegLnExpImp'`. We now go on to extend this criterion to deal with two-objective problems, i.e. our thrust vs. endurance trade-off.

3.3 Multi-objective expected improvement

For a single objective, we simply integrate $\phi(Y(\mathbf{x}))$ below the best value so far to obtain the probability of improvement. For two objectives we need to integrate a two-dimensional pdf:

$$\begin{aligned}\phi(Y_1, Y_2) = {} & \frac{1}{\widehat{s}_1(\mathbf{x})\sqrt{2\pi}} \exp\left[-\frac{(Y_1(\mathbf{x}) - \widehat{y}_1(\mathbf{x}))^2}{2\widehat{s}_1^2(\mathbf{x})}\right] \times \\ & \frac{1}{\widehat{s}_2(\mathbf{x})\sqrt{2\pi}} \exp\left[-\frac{(Y_2(\mathbf{x}) - \widehat{y}_2(\mathbf{x}))^2}{2\widehat{s}_2^2(\mathbf{x})}\right]\end{aligned} \tag{9}$$

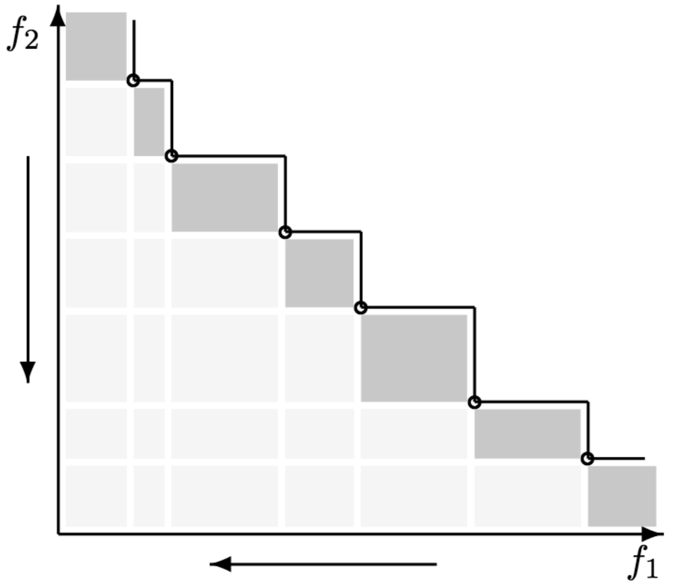

Fig. 5 A Pareto front and the areas where new points would augment the front (dark grey) and replace at least one point on the front (light grey). This figure is reproduced from Parr (2013).

(here we use subscripts to identify two uncorrelated Kriging models of our two objectives). For a two objective optimization there is not usually one best point to integrate below; there are a set of non-dominated solutions: a *Pareto front*, which we denote as $\mathbf{y}_1^\star$ and $\mathbf{y}_2^\star$ at locations $\mathbf{x}^\star$. Figure 5 is an example of a Pareto front and shows the areas for which Equation 9 must be integrated to find the probability of augmenting, improving, or augmenting OR improving on the current front. Furthermore, we need to calculate the centroid of this integral, $(\bar{Y}_1(\mathbf{x}), \bar{Y}_2(\mathbf{x}))$ to obtain the expected improvement:

$$E[I(\mathbf{x}^\star)] = P[I(\mathbf{x}^\star)] \left[\sqrt{(\bar{Y}_1(\mathbf{x}) - \mathbf{y}_1^\star)^2 + (\bar{Y}_2(\mathbf{x}) - \mathbf{y}_2^\star)^2} \right]_{\min}, \qquad (10)$$

where the $[.]_{\min}$ term refers to the minimum Euclidean distance between the centroid and a point on the Pareto front.

$P[I(\mathbf{x}^\star)]$ and $E[I(\mathbf{x}^\star)]$ are finally obtained after some rather tedious integration by parts, and the full expressions can be found in Parr (2013) (along with a review of other similar methods by Emmerich (2005) and Bautista (2009)). Following on from creating our two Kriging models of thrust and maximum muscle activity in Listing 3, Equation 10 can be evaluated using the script in Listing 5. This script goes on to maximize $E[I(\mathbf{x}^\star)]$ (in fact minimize $-\log_{10} E[I(\mathbf{x}^\star)]$) using a GA, and evaluate the objective functions at the resulting point (the Fourier coefficients calculated to give the best improvement over the current Pareto front).

Listing 5 *Matlab®* script to find the multi-objective expected improvement in the Kriging models of thrust and maximum muscle activity, maximise the expected improvement, add the corresponding point to the data set, and evaluate it.

```
% set option to calculate -log_10(EI)
ObjectiveInfo{1}.Option='NegLogExpImp';
% find -log_10(EI) at a point and return Pareto front
[EI,Py1,Py2,PX]=multiei(ones(1,15).*0.5,ObjectiveInfo);
figure
plot(ObjectiveInfo{1}.y,ObjectiveInfo{2}.y,'k.')
hold on
plot(Py1,Py2,'ko')
% set GA options as desired
options=gaoptimset('PopulationSize',50,'Generations',20);
% run GA, calling multiei
[varOpt,EIOpt]=ga(@(x) multiei(x,ObjectiveInfo),k,...
[],[],[],[],zeros(k,1),ones(k,1),[],options);
% add max(EI) point to data set
ObjectiveInfo{1}.X(end+1,:)=varOpt;
ObjectiveInfo{2}.X(end+1,:)=varOpt;
ObjectiveInfo{1}.y(end+1)=thrust(ObjectiveInfo{1}.X(end,:));
ObjectiveInfo{2}.y(end+1,1)=...
maxmuscleactivity(ObjectiveInfo{2}.X(end,:));
```

Having covered the methodology by which we optimize the objective functions, the next two sections consider the musculoskeletal and fluid modelling behind these functions. The results produced by running the scripts in this tutorial will then be examined in Section 6.

4 Musculoskeletal simulation

Here we use the *AnyBody Modelling System*™ (AMS) to simulate the biomechanics of the swimmer. For a prescribed motion with predefined forces applied, the AMS will estimate when and which muscles in the body are used to achieve this motion and how much force each muscle must produce. One method for determining realistic muscle loadings is by employing the min/-max criterion (Rasmussen, Damsgaard, and Voigt, 2001); the objective being to minimise the maximum muscle activity of all the muscles at each time step. A solution to which can be found using a computationally efficient linear optimisation process (Rasmussen, Damsgaard, and Voigt, 2001).

The basis of this criterion is that the body's central nervous system will strive to produce motion by activating each muscle in the most energy efficient manner. The min/max criterion has previously been used in other swimming related studies and has shown reasonable correlation to electromyography measurements (Nakashima et al., 2013). It is therefore assumed that this criterion can be seen as a surrogate for minimising the energy consumed for the given task and so for our simulations the maximum muscle activity is used as a surrogate for endurance.

The few examples of musculoskeletal models relating to human swimming in the literature include Nakashima et al. (2012) and Langholz, Westman, and

Karlsteen (2016) for example, however, for ease of integration with the optimisation process we have developed our own from the existing *FreePostureMove* model available from the *AnyBody managed model repository (AMMR).*

In this form of underwater swimming, the arms are raised above the swimmer's head inline with the long axis of the body – in effect increasing the water line length of the object moving through the water and helping to reduce hydrodynamic drag (Molland and Turnock, 2007). We assume that the arms remain in this position across all the fly-kick techniques we research and so the simulated energy consumption would not change across techniques. Since we are interested in comparing techniques (not in accurately predicting overall energy consumption) the arms can therefore be excluded from the musculoskeletal simulation, helping to reduce the complexity of the optimization process.

The resultant model consisted of 36 rigid body segments, connected by 137 joints and 519 muscles (158 in each leg). These 36 segments include all the constituent segments of the feet and spine. Even in the 'gold standard' of lab based optical motion capture there is insufficient kinematics to drive all segments. The model therefore estimates the kinematics of the unmeasured segments using those of the measured segments (Wong et al., 2006; Bassani et al., 2017).

The kinematics for the ankle and knee joint remained the same as the original mean kinematics for all simulations while the kinematics for the pelvis relative to the global reference frame, the hip and the pelvis-thorax angles were each refined as described in Section 2. The kinematics were assumed to be symmetrical for the left and right side of the body.

In recognition that many readers will not have access to the *AnyBody* software, we have provided in the online supplementary material a *Matlab*® p-code function `maxmuscleactivity()` which returns an approximation to the *AnyBody*® results. This function is in fact a Kriging model built from our database of results. We have validated the model with a bootstrapped 10-fold cross validation. Here we draw with replacement $1/10^{\mathrm{th}}$ of the data from that used to build the model, rebuild the model with these data removed, back-predict the removed data and calculate the r^2 value between the true data and model data. This process is repeated 1,000 times and yields a mean $\mu(r^2) = 0.9946$ and standard deviation $\sigma(r^2) = 0.0031$, indicating the model is reliably accurate at predicting maximum muscle activity.

5 Thrust simulation

As well as the kinematic components to the musculoskeletal model, it is also necessary to define any external loads. In underwater swimming, these forces consist of the hydrostatic forces (buoyancy) and hydrodynamic forces due to the movement of the body relative to the water. Gravitational forces are handled internally by the AMS.

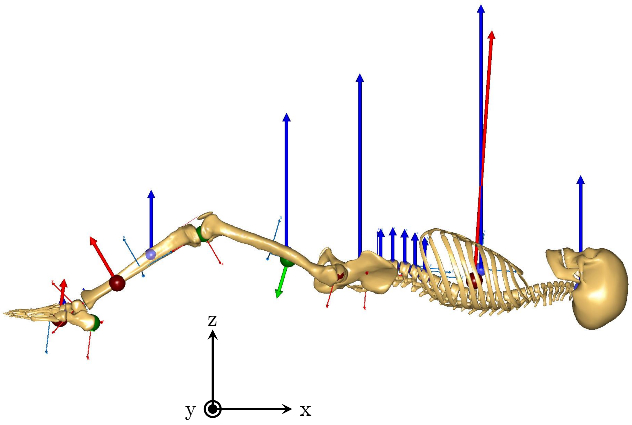

Fig. 6 A still image of the *AnyBody* model depicting the external forces. The vertical blue arrows illustrate the buoyancy forces applied at each segments' centre of mass. The green is the hydrodynamic force in the positive direction which is applied at the time dependant centre of pressure on each segment. Similarly the red arrow is the hydrodynamic force in the negative direction.

The hydrostatic forces are subject specific and therefore scaled in relation to the *AnyBody* model. Their direction is opposite to that of gravity, has magnitude proportional to the volume of the displaced water and its effort is applied to the centre of mass for each segment. Using the internal mass and density values in the *FreePostureMove* model, the magnitude of the buoyancy on segment *seg* is calculated as,

$$B_{seg} = \rho_f \; V_{seg} \; g \tag{11}$$

where ρ_f is the density of the fluid, V_{seg} the volume of the displaced fluid by the segment and g the acceleration due to gravity. The displaced fluid volume by each segment is calculated as,

$$V_{seg} = \frac{m_{seg}}{\rho_{seg}} \tag{12}$$

where m_{seg} and ρ_{seg} are the segment's mass and density, respectively. These hydrostatic forces are visualised by the large blue vertical arrows in Figure 5.

Dynamic pressure from the fluid also acts on each of the body segments. Because the segments are modelled as rigid bodies, it is possible to integrate the pressure over the segment and apply a point load at the centre of pressure perpendicular to the long-axis of the segment. The sign and magnitudes of these hydrodynamic point loads are visualised by the large red and green arrows in Figure 5. The centre of pressure at which these point loads act is depicted by the sphere of corresponding colour and is able to translate along the length of the segment at each time step.

Pedley and Hill (1999) review the mechanisms for predicting the muscle generated forces required in swimming fish. They identify that inertial forces dominate (large Reynolds number) with drag arising from viscous shear stress, pressure drag due to separation and due to the boundary layer growth along the body. Thrust is generated by the reactive inertia of the undulatory motion along the body creating a pressure difference either side of the body. The resultant flow field consists of regions of irrotational flow due to the added inertia of the fluid and vorticity created. The vorticity is shed from the body either at points of separation or at a sharp trailing edge. Fish are hydrodynamically streamlined with thin sharp edged fins unlike human swimmers whose toes are relatively blunt and are thus less effective in exploiting the propulsive benefits of generating trailing vortex structures.

Lighthill (1971) developed an analytical approach that captures the dominant inertial mechanism of large amplitude undulatory motion in generating forward thrust and this axial variation in transverse force. Due to its analytical approach it is computationally compact, typically two to three orders of magnitude faster than the vortex ring approach, to which it is compared by Pedley & Hill, and which primarily improves the vortex induced pressure field of the tail (of more significance in fish). Recent advances in computational power have allowed solutions of the Navier Stokes equations, typically using Reynolds averaging or Large Eddy simulation for undulatory motion. However, these computationally expensive approaches, at least six orders of magnitude more expensive than Lighthill (Molland and Turnock, 2007) are as yet unvalidated and only presented in a qualitative manner describing the flow field behaviour rather than giving a detailed breakdown of forces. The state-of-the-art in CFD is attempting to compute the forces on a forearm and hand during a front crawl stroke with qualitative agreement with experimental measurements (Samson et al., 2017), whereas progress is being made with validation of passive glide (Banks et al., 2014; Barbosa et al., 2017).

In seeking to understand the differences between different styles of underwater flykick, Lighthill's large amplitude analytical approach is applied in this work as it captures the dominant mechanism that generates a pressure difference against which the muscles work in a computationally efficient manner suitable for use in an optimisation study.

Webb et al. (2012) demonstrated the use of Lighthill's numerical theory developed for fish locomotion (Lighthill, 1970) to determine the forward thrust of a swimmer. As part of these calculations, the hydrodynamic side forces are also determined. The Lighthill approach (Lighthill, 1971) uses a momentum conservation method within a defined domain, ξ. This domain fully encapsulates the swimmer's motion but excludes the wake and hence is bound by plane Π, perpendicular to the tip of the swimmer's toes at all times. The swimmer is subdivided into n strips along their length, where each strip is described in global coordinates $x\left(a,t\right), z\left(a,t\right)$ in terms of parametric distance a, along the body at time t, where $0 < a < l$ and the constant l is the parametric length of

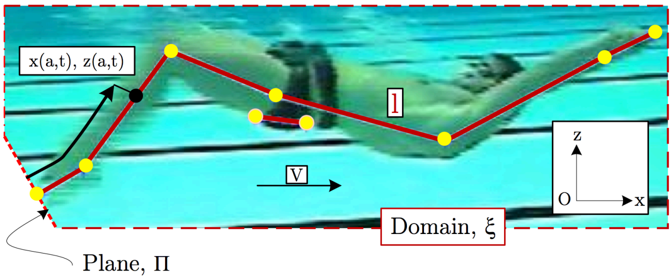

Fig. 7 The Lighthill reference system.

the body. This reference system is demonstrated in Figure 7. Lightill describes the rate of change of momentum in this control volume in terms of (i) the rate of change due to convection of momentum out of ξ across plane Π; (ii) plus the rate of change due to pressure forces acting across Π; and (iii) minus the reactive forces with which the fluid acts on the swimmer.

For prescribed kinematics, the thrust generated for propelling the swimmer through the water T and the side forces acting on the swimmer Q is expressed as,

$$(T,Q) = \left[mw\left(\frac{\partial z}{\partial t}, -\frac{\partial x}{\partial t}\right) - \left(\frac{1}{2}mw^2\left(\frac{\partial x}{\partial a}, \frac{\partial z}{\partial a}\right)\right)\right]_{a=0} - \frac{d}{dt}\int_0^l mw\left(-\frac{\partial z}{\partial a}, \frac{\partial x}{\partial a}\right). \tag{13}$$

where m is the mass per unit length ($m(a) = \frac{1}{4}\pi\rho s(a)^2$, where ρ is the water density and s the depth of the cross-section) and w is the velocity component perpendicular to the direction of parameter a.

For the experimental case, the forward speed of the swimmer was taken from the measured speed. A mean drag coefficient (C_D) was then calculated by,

$$\bar{T} = \frac{1}{2}\rho\bar{V}^2 SC_D \tag{14}$$

where $\bar{T}$ is the mean thrust, ρ the density of water, $\bar{V}$ the swimmer's mean forward speed and S the swimmer's frontal area (assumed proportional to kick amplitude (A) × breadth at the pelvis (B)).

To calculate the forward speed for all other kinematics, the thrust was first calculated using 13 and the same mean speed measured experimentally. An updated speed value was then calculated from 14 using the mean thrust and the same drag coefficient obtained from the experimental measurements. This process was itterated until the mean speed convereged ($< 0.003ms^{-1}$).

Using Lighthill's formula, we can then very quickly calculate both the hydrodynamic forces for the musculoskeletal model and determine the thrust produced; one of the objectives of this trade-off.

In the same manner as the *AnyBody* model at the end of Section 4, we have provided a supplementary material for the thrust model as a p-code

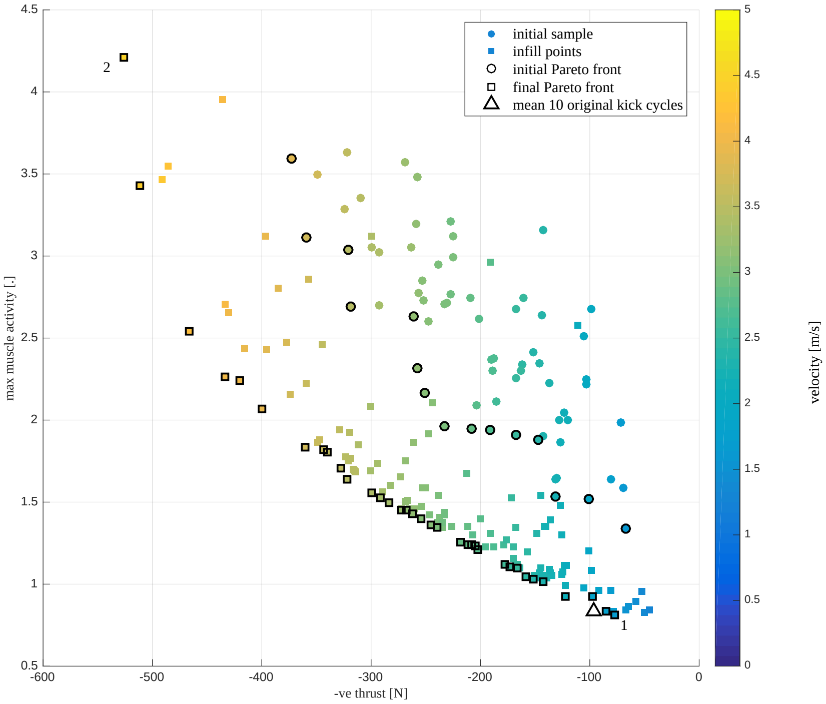

Fig. 8 The Pareto front showing the trade-off between the two objectives, thrust and maximum muscle activity. Point '1' is has the minimum maximum muscle activity and point '2' the maximum thrust.

function `thrust()` which returns an approximation to the thrust model. The bootstrapped 10-fold cross validation of the thrust model yields $\mu(r^2) = 0.9957$ and $\sigma(r^2) = 0.0048$

6 Results and discussion

Figure 8 shows the initial 75 point maximin Latin hypercube sample results as filled circles. Negative thrust is on the x-axis and maximum muscle activity on the y-axis, i.e. we wish to be in the south-west corner of the plot to produce more thrust for less power. The circles are coloured according to the velocity of the swimmer. We see immediately that more thrust means higher velocity (naturally, since we do not model the drag implications of different techniques), but that the relationship between max. muscle activity and velocity is non-linear on the axes we have used.

The best trade-off between thrust and max. muscle activity for the initial sample is depicted by black circles. Following 150 additional simulations (shown by filled squares), i.e. running Listing 5 150 times, the best trade-off points are improved to the black squares. These points indicate that the true Pareto front is likely to be a smooth, continuous line. As would be expected, the two objectives are in contention with each other; for an increased demand in thrust the maximum muscle activity displays a non-linear increase. Just as cost/performance Pareto front visualisations are useful in product design, here athletes and coaches might consult this Pareto front in order to examine techniques suitable for races of varying endurance.

The kinematics that produced the minimum maximum muscle activity (labelled '1' in Figure 8) and those that delivered the maximum thrust (labelled '2') are overlaid with the original kinematics in Figure 9. For each joint, both objectives' results differed from the original experimental data of the elite swimmer. In the hip joint, the kinematics for greatest thrust have twice the frequency of the original kinematics including a greater peak angle. The minimum maximum muscle activity case, conversely, has only one peak per cycle – similar to the original kinematics in terms of shape and range of motion, but at a higher flexion angle. For the pelvis-thorax angle the maximum thrust and minimum maximum muscle activity results produce kinematics with two distinct oscillations per cycle in contrast to the experimental data, and with roughly half the range of motion. A shift in flexion angle is the only difference between maximum thrust and minimum maximum muscle activity. The same double oscillation of the maximum thrust case is also seen in the pelvic-pitch. The minimum maximum muscle activity case also has a double oscillation, but at a reduced amplitude, and out of phase with the maximum thrust case.

The mean thrust and maximum muscle activity of the 10 original kick cycles is shown as a triangle, and lies on the Pareto front near to the minimum maximum muscle activity end. While the optimization process has found a range of techniques with optimal thrust/energy tradeoff, it has not actually improved on the swimmer's current technique for that given thrust and muscle activity. This is somewhat reassuring insofar as the optimizer seems not to have exploited assumptions/loopholes in the analyses to find unrealistic objective function values. However, the higher thrust techniques at the other end of the Pareto front may not be attainable.

By basing our kick parameterisation on Fourier series fitted to experimental data, we have assumed that all motions investigated could be implemented by a swimmer. This may not be the case, as one joint angle will naturally influence the range of motion in others. Ensuring all motions are physically realisable would require us to work our way up/down the kinematic chain from the pelvis, ensuring distal joint ranges of motion are compatible with the proximal joint (a Markov chain process).

Further enhancements to the study presented here might naturally include improving the fidelity of both the fluid dynamic and musculoskeletal modelling, though this would come at increased computational expense, reducing the number of techniques that could be investigated and so may not necessarily improve the results. A more fruitful avenue of research may be to work with swimmers/coaches to implement the optimal techniques found and provide some validation for the results presented.

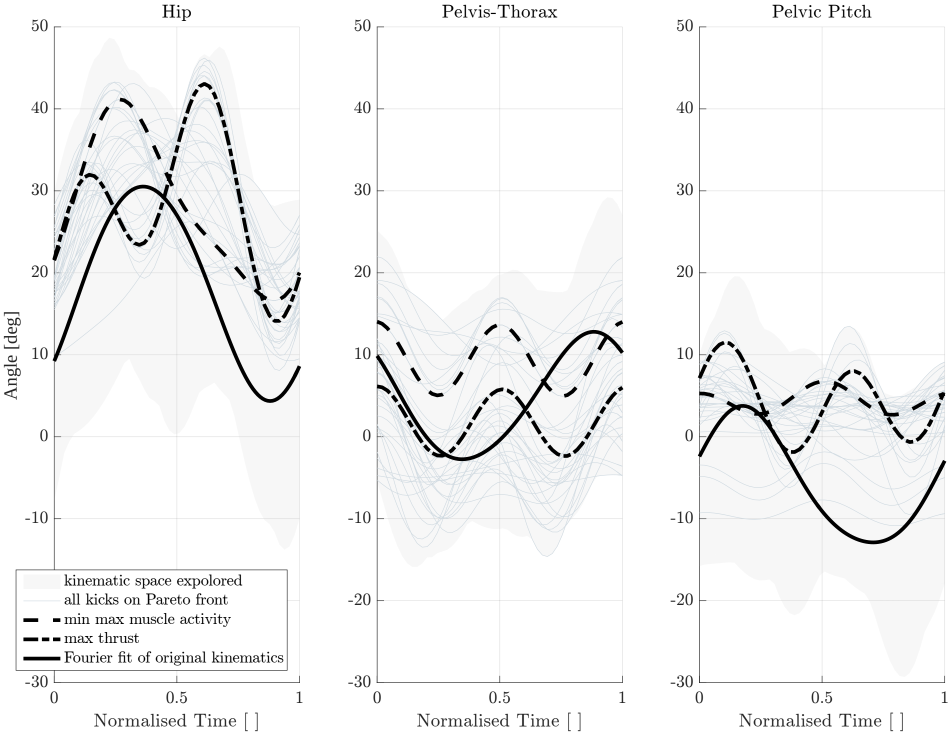

Fig. 9 The kinematics for one kick-cycle.The dashed line represent the optimal kinematics for the objecting of minimising the mean muscle activity and the dashed-dotted line identifies the kinematics that maximise the mean thrust.

References

Banks, J, MC James, SR Turnock, and DA Hudson. 2014. “An analysis of a swimmers passive wave resistance using experimental data and CFD simulations.” .

Barbosa, Tiago M, Rui Ramos, António J Silva, and Daniel A Marinho. 2017. “Assessment of passive drag in swimming by numerical simulation and analytical procedure.” *Journal of Sports Sciences* 1–7.

Bassani, Tito, Elena Stucovitz, Zhihui Qian, Matteo Briguglio, and Fabio Galbusera. 2017. “Validation of the AnyBody full body musculoskeletal model in computing lumbar spine loads at L4L5 level.” *Journal of Biomechanics* 58: 89–96.

Bautista, Dianne C. 2009. “A sequential design for approximating the pareto front using the expected pareto improvement function.” Ph.D. thesis. The Ohio State University.

Emmerich, Michael T.M. 2005. “Single-and multi-objective evolutionary design optimization assisted by gaussian random field metamodels.” Ph.D. thesis. University of Dortmund.

Forrester, Alexander I.J., and Andy J. Keane. 2009. "Recent advances in surrogate-based optimization." *Progress in Aerospace Sciences* 45 (1-3): 50–79.

Forrester, Alexander I.J., Andras Sóbester, and Andy J. Keane. 2008. *Engineering Design via Surrogate Modelling: A Practical Guide.* John Wiley & Sons Ltd.

Ginsbourger, David, Nicolas Durrande, and Olivier Roustant. 2013. "Kernels and designs for modelling invariant functions: From group invariance to additivity." *Advances in Model-Oriented Design and Analysis* 107–115.

Jones, D. R. 2001. "A Taxonomy of Global Optimization Methods Based on Response Surfaces." *Journal of Global Optimization* 21: 345–383.

Jones, Donald R., Matthias Schonlau, and William J. Welch. 1998. "Efficient Global Optimization of Expensive Black-Box Functions." *Journal of Global Optimization* 13 (4): 455–492.

Langholz, Janna Brit, Gunnar Westman, and Magnus Karlsteen. 2016. "ScienceDirect Musculoskeletal modelling in sports -evaluation of different software tools with focus on swimming." *Procedia Engineering* 147: 281–287.

Lighthill, M. J. 1970. "Aquatic animal propulsion of high hydromechanical efficiency." *Journal of Fluid Mechanics* 44 (02): 265–301.

Lighthill, M. J. 1971. "Large-Amplitude Elongated-Body Theory of Fish Locomotion." *Proceedings of the Royal Society of London. Series B, Biological Sciences* 179 (1055): 125–138.

Loeppky, Jason L., Jerome Sacks, and William J. Welch. 2009. "Choosing the Sample Size of a Computer Experiment: A Practical Guide." *Technometrics* 51 (4): 366–376.

Molland, Anthony F., and Stephen R. Turnock. 2007. *Marine rudders and control surfaces.* first edit ed. Butterworth-Heinemann.

Morris, M D, and T J Mitchell. 1995. "Exploratory Designs for Computer Experiments." *J. Statist. Plann. Inference* 43 (1995): 381–402.

Nakashima, Motomu, Takahiro Hasegawa, Seiji Kamiya, and Hideki Takagi. 2013. "Musculoskeletal Simulation of the Breaststroke." *Journal of Biomechanical Science and Engineering* 8 (2): 152–163.

Nakashima, Motomu, Shun Maeda, Takahiro Miwa, and Hiroshi Ichikawa. 2012. "Optimizing Simulation of the Arm Stroke in Crawl Swimming Considering Muscle Strength Characteristics of Athlete Swimmers." *Journal of Biomechanical Science and Engineering* 7 (2): 102–117.

Parr, James M. 2013. "Improvement Criteria for Constraint Handling and Multiobjective Optimization." Ph.D. thesis. University of Southampton.

Pedley, T.J., and S.J. Hill. 1999. "Large-amplitude undulatory fish swimming: fluid mechanics coupled to internal mechanics." *The Journal of Experimental Biology* 202 (23): 3431–3438.

Phillips, Christopher W.G. 2013. "Analysis of human underwater undulatory swimming using musculoskeletal modelling." Ph.D. thesis. University of Southampton.

Phillips, Christopher W.G., Alexander I.J. Forrester, Dominic A. Hudson, and Stephen R. Turnock. 2014. "Comparison of Kinematic Acquisition Methods for Musculoskeletal Analysis of Underwater Flykick." *Procedia Engineering* 72 (0): 56–61.

Rasmussen, John, Michael Damsgaard, and M Voigt. 2001. "Muscle recruitment by the min/max criterion – a comparative numerical study.." *Journal of biomechanics* 34 (3): 409–15.

Roustant, Olivier, David Ginsbourger, and Yves Deville. 2012. "DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization." *Journal of Statistical Software* 51 (1): 1–55.

Samson, Mathias, Anthony Bernard, Tony Monnet, Patrick Lacouture, and Laurent David. 2017. "Unsteady computational fluid dynamics in front crawl swimming." *Computer Methods in Biomechanics and Biomedical Engineering* 20 (7): 783–793.

Sóbester, András, Stephen J. Leary, and Andy J. Keane. 2005. "On the Design of Optimization Strategies Based on Global Response Surface Approximation Models." *Journal of Global Optimization* 33 (1): 31–59.

Webb, Angus P., Christopher W.G. Phillips, Dominic A. Hudson, and Stephen R. Turnock. 2012. "Can Lighthill's Elongated Body Theory Predict Hydrodynamic Forces in Underwater Undulatory Swimming?." *Procedia Engineering* 34 (0): 724–729.

Wong, Kris WN, Keith DK Luk, John CY Leong, S F Wong, and Kenneth KY Wong. 2006. "Continuous dynamic spinal motion analysis." *Spine* 31 (4): 414–419.