INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2003; **57**:2205–2219 (DOI: 10.1002/nme.762)

Computational design of swarms

T. I. Zohdi*,†

Department of Mechanical Engineering, University of California, 6195 Etcheverry Hall, Berkeley, CA 94720-1740, U.S.A.

SUMMARY

In this work, a method for the rapid computational design and simulation of multiparticle swarms is developed. Specifically, a non-convex optimization strategy is constructed, based on genetic algorithms, to design desired swarm-like behaviour. To allow rapid evaluation of various swarm design performances, a temporally-adaptive iterative scheme is developed to determine the positions, velocities and accelerations of members of the swarm. The overall purpose of such an approach is to facilitate rapid decision making for the dynamics of large numbers of interacting objects, whose goal is to reach a target guarded by obstacles in a minimum amount of time. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: swarms; optimization

1. INTRODUCTION

It has long been recognized that interactive cooperative behaviour within biological groups or 'swarms' is advantageous in avoiding predators or, vice versa, in capturing prey. For example, one of the primary advantages of a swarm-like decentralized decision making structure is that there is no leader and thus the vulnerability of the swarm is substantially reduced. Furthermore, the decision making is relatively simple and rapid for each individual, however, the aggregate behaviour of the swarm can be quite sophisticated. Although the modelling of swarm-like behaviour has biological research origins, dating at least back to Breder [1], it can be treated as a purely multiparticle dynamical system, where the communication between swarm members is modelled via interaction forces. It is commonly accepted that a central characteristic of swarm-like behaviour is the tradeoff between long-range interaction and short-range repulsion between individuals. Models describing clouds or swarms of particles, where their interaction is constructed from attractive and repulsive forces, dependent on the relative

*Correspondence to: T. I. Zohdi, Department of Mechanical Engineering, University of California, 6195 Etcheverry Hall, Berkeley, CA 94720-1740, U.S.A.
†E-mail: zohdi@newton.berkeley.edu

Received 23 August 2002
Revised 2 December 2002
Accepted 23 December 2002

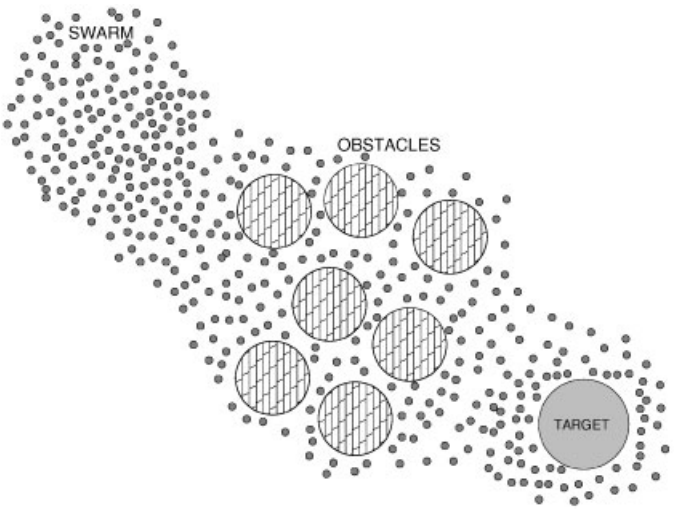

Figure 1. Desired swarm-like behaviour: avoiding obstacles and reaching a target.

distance between individuals, are commonplace. For reviews, see Reference [2–4]. Investigation of a closely related class of systems, platooning of vehicles in automated highway systems and unmanned autonomous vehicles (UAVs), where the swarm members each have their own propulsion system, can be found in, among others, References [5–9] and recently in Reference [10]. Of course the field is quite large and encompasses a wide variety of applications, for example, the behaviour of bird flocks, fish schools, flow of traffic and crowds of human beings, to name a few. Loosely speaking, swarm analyses are concerned with the resulting aggregate behaviour of groups of simple members. Recent broad overviews can be found in References [11, 12].

The steady growth in computing power has raised the possibility of direct high speed simulation of swarm movement. A certain class of rapid solution methods for groups of particles undergoing strongly coupled multifield thermo-chemo-mechanical interaction, based on adaptive staggering algorithms, have recently been developed in Reference [13] and modified in Reference [14] for particles in granular flows related to the aggregation of micron-sized granular dust particles in gaseous accretion disks. For reviews of the physical underpinnings of such processes, see References [15–23].

An important aspect of the ability to perform rapid computational simulation of swarm dynamics is the possibility to *optimize the interaction* of the swarm members so that they can achieve a desired overall goal (Figure 1). In order to achieve this, in this work, a statistical genetic algorithm is developed to handle associated objective functions which depend in a non-convex and non-differentiable manner on the free variables in the system. Such approaches require large numbers of swarm-design evaluations, partly due to the stochastic nature of such problems, which require ensemble averaging, therefore fast simulation techniques are developed here based on approaches found in References [13, 14]. Both the optimization and rapid simulation of swarms are themes that are central to the present work.

2. CONSTRUCTION OF AN INTERACTING MULTIPARTICLE SWARM

In the analysis to follow, we treat the swarm members as point masses, i.e. we ignore their dimensions.[‡] For each swarm member (N in total) the equations of motion at time t are

$$m_i \ddot{\mathbf{r}}_i = \boldsymbol{\Psi}_i(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_n) \tag{1}$$

where $\boldsymbol{\Psi}$ represents the forces of interaction between swarm member i and the target, obstacles and other swarm members. We consider the following decomposition:

$$\boldsymbol{\Psi}_i = \boldsymbol{\Psi}_i^{M-M} + \boldsymbol{\Psi}_i^{M-T} + \boldsymbol{\Psi}_i^{M-O} \tag{2}$$

where between swarm members

$$\boldsymbol{\Psi}_i^{M-M} = \sum_{k \neq i}^{n} \left(\left(\underbrace{\alpha_1^{M-M} \|\mathbf{r}_i - \mathbf{r}_k\|^{\beta_1^{M-M}}}_{\text{attraction}} - \underbrace{\alpha_2^{M-M} \|\mathbf{r}_i - \mathbf{r}_k\|^{-\beta_2^{M-M}}}_{\text{repulsion}} \right) \underbrace{\frac{\mathbf{r}_i - \mathbf{r}_k}{\|\mathbf{r}_i - \mathbf{r}_k\|}}_{\text{unit vector}} \right) \tag{3}$$

where $\|\cdot\|$ represents the euclidean norm in R^3, while between the swarm members and the target we have

$$\boldsymbol{\Psi}_i^{M-T} = \left(\alpha^{M-T} \|\mathbf{r}_* - \mathbf{r}_i\|^{\beta^{M-T}} \right) \frac{\mathbf{r}_* - \mathbf{r}_i}{\|\mathbf{r}_* - \mathbf{r}_i\|} \tag{4}$$

and for the repulsion between swarm members and the obstacles

$$\boldsymbol{\Psi}_i^{M-O} = -\sum_{j=1}^{q} \left(\left(\alpha^{M-O} \|\mathbf{r}_{Oj} - \mathbf{r}_i\|^{-\beta^{M-O}} \right) \frac{\mathbf{r}_{Oj} - \mathbf{r}_i}{\|\mathbf{r}_{Oj} - \mathbf{r}_i\|} \right) \tag{5}$$

where q is the number of obstacles and where all of the (design) parameters, αs and βs, are non-negative.

Remark

One can describe the relative contributions of repulsion and attraction between members of the swarm by considering an individual pair in static equilibrium

$$\boldsymbol{\Psi}_i^{M-M} = \left(\alpha_1^{M-M} \|\mathbf{r}_i - \mathbf{r}_k\|^{\beta_1^{M-M}} - \alpha_2^{M-M} \|\mathbf{r}_i - \mathbf{r}_k\|^{-\beta_2^{M-M}} \right) \frac{\mathbf{r}_i - \mathbf{r}_k}{\|\mathbf{r}_i - \mathbf{r}_k\|} = \mathbf{0} \tag{6}$$

This characterizes a separation length scale describing the tendency to cluster or spread apart

$$\|\mathbf{r}_i - \mathbf{r}_k\| = \left(\frac{\alpha_2^{M-M}}{\alpha_1^{M-M}} \right)^{\frac{1}{\beta_1^{M-M} + \beta_2^{M-M}}} \stackrel{\text{def}}{=} \rho^{M-M} \tag{7}$$

We remark that one could have moving targets and obstacles, as well as attractive forces between the swarm and the obstacles and repulsive forces from the targets. Adding attractive

[‡]The swarm member centres, which are initially non-intersecting, cannot intersect later due to the singular repulsion terms.

forces from the obstacles and repulsive forces from the targets makes sense for some applications, for example in traffic flow, where one does not want the vehicle to hit the target, although we did not consider such cases in the present work.

3. AN ITERATIVE SOLUTION SCHEME

In order to drive the optimization process which follows later, the simulation of a swarm must be solved in a relatively rapid and robust manner. After time discretization of acceleration term in the equations of motion for a swarm member $\ddot{\mathbf{r}}^{L+1} \approx (\mathbf{r}^{L+1} - 2\mathbf{r}^{L} + \mathbf{r}^{L-1})/(\Delta t)^2$, where $\mathbf{r}^{L+1} \stackrel{\text{def}}{=} \mathbf{r}(t+\Delta t)$, one can arrive at the following abstract form, $\mathbf{A}(\mathbf{r}^{L+1}) = \mathbf{F}$. It is convenient to write

$$\mathbf{A}(\mathbf{r}^{L+1}) - \mathbf{F} = \mathbf{G}(\mathbf{r}^{L+1}) - \mathbf{r}^{L+1} + \mathbf{d} = \mathbf{0} \tag{8}$$

A straightforward iterative scheme is

$$\mathbf{r}^{L+1,K} = \mathbf{G}(\mathbf{r}^{L+1,K-1}) + \mathbf{d} \tag{9}$$

where $K = 1, 2, 3, \ldots$ is the index of iteration within time step $L+1$. The convergence of such a scheme is dependent on the behaviour of $\mathbf{G}$. Namely, a sufficient condition for convergence is that $\mathbf{G}$ is a contraction mapping for all $\mathbf{r}^{L+1,K}$, $K = 1, 2, 3 \ldots$ Accordingly, we define the error as $\mathbf{e}^{L+1,K} = \mathbf{r}^{L+1,K} - \mathbf{r}^{L+1}$. A necessary restriction for convergence is iterative self consistency, i.e. the exact solution must be represented by the scheme $\mathbf{G}(\mathbf{r}^{L+1}) + \mathbf{d} = \mathbf{r}^{L+1}$. Enforcing this restriction, a sufficient condition for convergence is the existence of a contraction mapping

$$\|\mathbf{e}^{L+1,K}\| = \|\mathbf{r}^{L+1,K} - \mathbf{r}^{L+1}\| = \|\mathbf{G}(\mathbf{r}^{L+1,K-1}) - \mathbf{G}(\mathbf{r}^{L+1})\| \leqslant \eta^{L+1,K} \|\mathbf{r}^{L+1,K-1} - \mathbf{r}^{L+1}\| \tag{10}$$

where, if $\eta^{L+1,K} < 1$ for each iteration K, then $\mathbf{e}^{L+1,K} \to \mathbf{0}$ for any arbitrary starting value $\mathbf{r}^{L+1,K=0}$ as $K \to \infty$. The type of contraction condition discussed is sufficient, but not necessary, for convergence. In order to control convergence we modify the discretization[§]

$$\ddot{\mathbf{r}}_i^{L+1} = \frac{\dot{\mathbf{r}}_i^{L+1} - \dot{\mathbf{r}}_i^{L}}{\Delta t} = \frac{\frac{\mathbf{r}_i^{L+1} - \mathbf{r}_i^{L}}{\Delta t} - \dot{\mathbf{r}}_i^{L}}{\Delta t} = \frac{\mathbf{r}_i^{L+1} - \mathbf{r}_i^{L}}{(\Delta t)^2} - \frac{\dot{\mathbf{r}}_i^{L}}{\Delta t} = \frac{\mathbf{r}_i^{L+1}}{(\Delta t)^2} - \frac{\mathbf{r}_i^{L}}{(\Delta t)^2} - \frac{\dot{\mathbf{r}}_i^{L}}{\Delta t} \tag{11}$$

and write the following

$$\mathbf{r}_i^{L+1,K} = \underbrace{\frac{(\Delta t)^2}{m_i} \left(\boldsymbol{\Psi}_i \left(\mathbf{r}_1^{L+1,K-1}, \mathbf{r}_2^{L+1,K-1}, \ldots, \mathbf{r}_n^{L+1,K-1} \right) \right)}_{G(\mathbf{r}^{L+1,K-1})} + \mathbf{r}_i^{L} + \Delta t \dot{\mathbf{r}}_i^{L} \tag{12}$$

Therefore, we see that the eigenvalues of $\mathbf{G}$ are (1) directly dependent on the strength of the interaction forces, (2) inversely proportional to the masses of the swarm members and (3) directly proportional to $(\Delta t)^2$. Therefore, if convergence is slow within a time step, the time step is reduced by an appropriate amount to increase the rates of convergence.

[§]This collapses to a stencil of $\ddot{\mathbf{r}}^{L+1} = (\mathbf{r}^{L+1} - 2\mathbf{r}^{L} + \mathbf{r}^{L-1})/(\Delta t)^2$, when the time step size is uniform.

Classical solution methods require $\mathcal{O}(N^3)$ operations, whereas iterative schemes, such as the one presented, typically require order N^q, where $1 \leqslant q \leqslant 2$. For details see Reference [24]. Also, such solvers are highly advantageous since solutions to previous time steps can be used as the first guess to accelerate the solution procedure.

In order to control the convergence of the iterative scheme, we follow an approach in Reference [13] originally developed for thermo-chemical multifield problems where (1) one approximates $\eta^{L+1,K} \approx \theta(\Delta t)^p$ (θ is a constant) and (2) one approximates the error within an iteration to behave according to $(\theta(\Delta t)^p)^K \|e^{L+1,0}\| = \|e^{L+1,K}\|$, $K = 1, 2, \ldots$, where $\|e^{L+1,0}\|$ is the initial norm of the iterative error and θ is a function intrinsic to the system.¶ Our goal is to meet an error tolerance in exactly a preset number of iterations. To this end, one writes this in the following approximate form, $(\theta(\Delta t_{\text{tol}})^p)^{L+1,K_d} \|e^{L+1,0}\| = \text{TOL}$, where TOL is a tolerance and where K_d is the number of desired iterations.‖ If the error tolerance is not met in the desired number of iterations, the contraction constant $\eta^{L+1,K}$ is too large. Accordingly, one can solve for a new smaller step size, under the assumption that θ is constant,

$$\Delta t_{\text{tol}} = \Delta t \left(\frac{\left(\frac{\text{TOL}}{\|e^{L+1,0}\|} \right)^{\frac{1}{pK_d}}}{\left(\frac{\|e^{L+1,K}\|}{\|e^{L+1,0}\|} \right)^{\frac{1}{pK}}} \right) \tag{13}$$

The assumption that θ is constant is not critical, since the time steps are to be recursively refined and unrefined. Clearly, the expression in Equation (13) can also be used for time step enlargement, if convergence is met in less than K_d iterations. An overall algorithm employing such relations for temporal adaptivity, within a genetic algorithm for swarm design, is given in the next section.

Remarks

A recursive iterative scheme of Jacobi-type, where the updates are made only after one complete iteration, was considered here only for algebraic simplicity. The Jacobi method is easier to address theoretically, thus it is used for proof of convergence, while the Gauss–Seidel type method, which involves immediately using the most current values, when they become available, is used at the implementation level. As is well-known that, under relatively general conditions, if the Jacobi method converges, the Gauss–Seidel method converges at a faster rate, while if the Jacobi method diverges, the Gauss–Seidel method diverges at a faster rate References [25–28].

4. OPTIMAL DESIGN OF A SWARM VIA GENETIC ALGORITHMS

We now consider inverse problems whereby the coefficients in the interaction forces are sought, the αs and βs, which deliver desired swarm-like behaviour by minimizing a normalized cost function representing (1) the time it takes for the swarm members to get to the target

¶For the class of problems under consideration, $p = 2$.

‖Typically, K_d is chosen to be between five and ten iterations.

and (2) the distance of the swarm members away from the target:

$$\Pi = \frac{\left(\int_0^{\mathscr{T}} \sum_{i=1}^{n} \|\mathbf{r}_i - \mathbf{r}_*\| \, \mathrm{d}t \right)}{\mathscr{T} \sum_{i=1}^{n} \|\mathbf{r}_i(t=0) - \mathbf{r}_*\|} \tag{14}$$

where total simulation time is $\mathscr{T} = 1$; $\boldsymbol{\alpha}^- \leqslant \boldsymbol{\alpha} \leqslant \boldsymbol{\alpha}^+$ and $\boldsymbol{\beta}^- \leqslant \boldsymbol{\beta} \leqslant \boldsymbol{\beta}^+$; $\mathbf{r}_*$ is the position of the target; $\boldsymbol{\alpha}^-$, $\boldsymbol{\alpha}^+$, $\boldsymbol{\beta}^-$ and $\boldsymbol{\beta}^+$ are the lower and upper limits coefficients in the interaction forces. As a side condition, $\forall\ t$, $\forall\ r_{Oj}$ and $\tau < \mathscr{T}$, if $\|\mathbf{r}_i(t=\tau) - \mathbf{r}_{Oj}\| \leqslant R$ then $\mathbf{r}_i = \mathbf{r}_i(t=\tau)$, $\forall\ t \geqslant \tau$, where the unilateral condition represents the effect of being near a 'destructive' obstacle. The swarm member is stopped in the position where it enters the 'radius of destruction' (R). Therefore, the swarm performance (Π) is severely penalized if it loses members to the obstacles. With respect to the minimization of Equation (14), classical gradient-based deterministic optimization techniques are not robust, due to difficulties with objective function non-convexity and non-differentiability. These difficulties can be circumvented by the use of a certain class of non-derivative search methods, usually termed 'genetic' algorithms (GA). There are a variety of such methods, which employ concepts of species evolution, such as reproduction, mutation and crossover. For reviews see References [11, 29].

In this work, employing basic GA concepts, an algorithm has been developed to treat non-convex inverse problems of swarm design. The central idea is that the swarm parameters form a genetic string and a survival of the fittest algorithm is applied to a population of such strings. The overall process is (I) A population (S) of different designs are generated at random with the design space, represented by a ('genetic') string of the design (N) parameters, (II) the performance of each design is tested, (III) the designs are ranked from top to bottom according to their performance, (IV) the best designs are mated pairwise producing two offspring, i.e. each best pair exchanges information by taking random convex combinations of the design components of the parents' genetic strings and (V) the worst performing genetic strings are eliminated, new replacement designs (strings) are introduced into the remaining population of best performing genetic strings and the process (I)–(V) is then repeated. An implementation of such ideas is as in Box 1.

Remark I

The term 'fitness' of a genetic string in this algorithm indicates the value of the objective function. The most fit genetic string is the one with the smallest objective function. The retention of the top old fit genetic strings is critical, since, referring to Figure 2,

one sees that, if the objective functions are highly non-convex (the present case), there exists a clear possibility that the inferior offspring will replace superior parents. When the top parents are retained, the minimization of the cost function is guaranteed to be monotone (guaranteed improvement) with increasing generations. There is no guarantee of successive improvement if the top parents are not retained, even though non-retention of parents allows more newer (possibly better) genetic strings to be evaluated in the next generation. Numerical studies conducted by the author imply, for sufficiently large populations, that the benefits of parent retention outweigh this advantage and any disadvantages of 'inbreeding', i.e. a stagnant population. For more details on this so-called 'inheritance property' see References [29] or [11]. In the presented algorithm, inbreeding is mitigated since, with each new generation, new design, selected

STEP 1: (OUTER LOOP) RANDOMLY SELECT N STARTING GENETIC STRINGS $\mathbf{\Lambda}^I$, $(I=1,\ldots,S)$:

$$\mathbf{\Lambda}^I \stackrel{\text{def}}{=} \{\Lambda_1^I, \Lambda_2^I, \Lambda_3^I, \Lambda_4^I, \Lambda_5^I, \Lambda_6^I, \Lambda_7^I, \Lambda_8^I\}$$

$$= \{(\alpha^{M-M})_1^I, (\alpha^{M-M})_2^I, (\alpha^{M-T})^I, (\alpha^{M-O})^I, (\beta^{M-M})_1^I, (\beta^{M-M})_2^I, (\beta^{M-T})^I, (\beta^{M-O})^I\}$$

STEP 2: COMPUTE FITNESS $(\Pi(\mathbf{\Lambda}^I))$ OF EACH GENETIC STRING: $(I=1,\ldots,S)$

(2.1.a) (INNER LOOP) INTERNAL FIXED POINT ITERATION: (SET $i=1$ AND $K=0$):
(2.1.b) IF $i>N$ THEN GO TO (2.2.a)
(2.1.c) IF $i\leqslant N$ THEN SOLVE $m_i\ddot{\mathbf{r}}_i^{L+1,K} = \mathbf{\Psi}_i(\mathbf{r}_1^{L+1,K-1}, \mathbf{r}_2^{L+1,K-1}, \ldots, \mathbf{r}_n^{L+1,K-1})$

(1) $\mathbf{r}_i^{L+1,K} = \frac{(\Delta t)^2}{m_i}\left(\sum_{j\neq i}^{n} \mathbf{\Psi}_i\left(\mathbf{r}_1^{L+1,K-1}, \mathbf{r}_2^{L+1,K-1}, \ldots, \mathbf{r}_n^{L+1,K-1}\right)\right) + \mathbf{r}_i^L + \Delta t\dot{\mathbf{r}}_i^L$

(2) COMPUTE ITERATION DIFFERENCE: $\gamma_i^{L+1,K} \stackrel{\text{def}}{=} \|\mathbf{r}_i^{L+1,K} - \mathbf{r}_i^{L+1,K-1}\|$
(3) UPDATE POSITION: $\mathbf{r}_i^{L+1,K-1} = \mathbf{r}_i^{L+1,K}$
(4) GO TO (2.1.b) AND NEXT SWARM MEMBER $(i=i+1)$

(2.2.a) ERROR MEASURES:

(1) $\|e^{L+1,K}\| \stackrel{\text{def}}{=} \frac{\sum_{i=1}^{n}\gamma_i^{L+1,K}}{\sum_{i=1}^{n}\|\mathbf{r}_i^{L+1,K}\|}$, (2) $\xi^{L+1,K} \stackrel{\text{def}}{=} \frac{\|e^{L+1,K}\|}{TOL}$, (3) $\phi^{L+1,K} \stackrel{\text{def}}{=} \left(\frac{\left(\frac{TOL}{\|e^{L+1,0}\|}\right)^{\frac{1}{pK_d}}}{\left(\frac{\|e^{L+1,K}\|}{\|e^{L+1,0}\|}\right)^{\frac{1}{pK}}}\right)$

(2.2.b) IF TOLERANCE MET $(\xi^{L+1,K}\leqslant 1)$ AND $K<K_d$ THEN:

(1) CONSTRUCT NEW TIME STEP: $\Delta t = \phi^{L+1,K}\Delta t$ AND $t = t+\Delta t$
(2) SELECT MINIMUM: $\Delta t = MIN(\Delta t^{DISCRETE}, \Delta t)$
$(\Delta t^{DISCRETE} =$ UPPER TIME STEP DISCRETIZATION SIZE LIMIT$)$
(3) $\Pi(\mathbf{\Lambda}^I) = \dfrac{\left(\int_0^{\mathscr{T}}\sum_{i=1}^{n}\|\mathbf{r}_i-\mathbf{r}_*\|\,\mathrm{d}t\right)}{\mathscr{T}\sum_{i=1}^{n}\|\mathbf{r}_i(t=0)-\mathbf{r}_*\|}$

(ENFORCE: $\forall t$ AND $\forall r_{Oj}$ AND $\tau<\mathscr{T}$ IF $\|\mathbf{r}_i(t=\tau)-\mathbf{r}_{Oj}\|\leqslant R$ THEN $\mathbf{r}_i=\mathbf{r}_i(t=\tau)$, $\forall t\geqslant\tau$)
(4) INCREMENT TIME: IF $t<\mathscr{T}$ THEN $t=t+\Delta t$ AND GO TO (2.1)
(5) IF I = S GO TO STEP 3. IF I < S GO TO STEP 2.

(2.2.c) IF TOLERANCE NOT MET $(\xi^{L+1,K}>1)$ AND $K=K_d$ THEN:

(1) STEP BACK: $t=t-\Delta t$
(2) CONSTRUCT NEW TIME STEP: $\Delta t = \phi^{L+1,K}\Delta t$
(3) RESTART AT TIME t AND GO TO (2.1.a)

STEP 3: RANK THE GENETIC STRINGS, $\mathbf{\Lambda}^I$ $(I=1,\ldots,S)$
STEP 4: MATE NEAREST PAIRS (PRODUCE OFFSPRING FIGURE 2) $(I=1,\ldots,S)$

$$\boldsymbol{\lambda}^I \stackrel{\text{def}}{=} \Phi^{(1)}\mathbf{\Lambda}^I + (1-\Phi^{(1)})\mathbf{\Lambda}^{I+1} \qquad \boldsymbol{\lambda}^{I+1} \stackrel{\text{def}}{=} \Phi^{(2)}\mathbf{\Lambda}^I + (1-\Phi^{(2)})\mathbf{\Lambda}^{I+1}$$

$0\leqslant\Phi^{(1)},\Phi^{(2)} = RAND\leqslant 1$ (DIFFERENT RANDOM NUMBERS FOR EACH COMPONENT)

STEP 5: KILL OFF BOTTOM M < S STRINGS. OPTIONAL: KEEP TOP K PARENTS
STEP 6: REPEAT WITH TOP GENE POOL PLUS M NEW ONES: $\mathbf{\Lambda}^I = \boldsymbol{\lambda}^I$, $(I=1,\ldots,S)$.

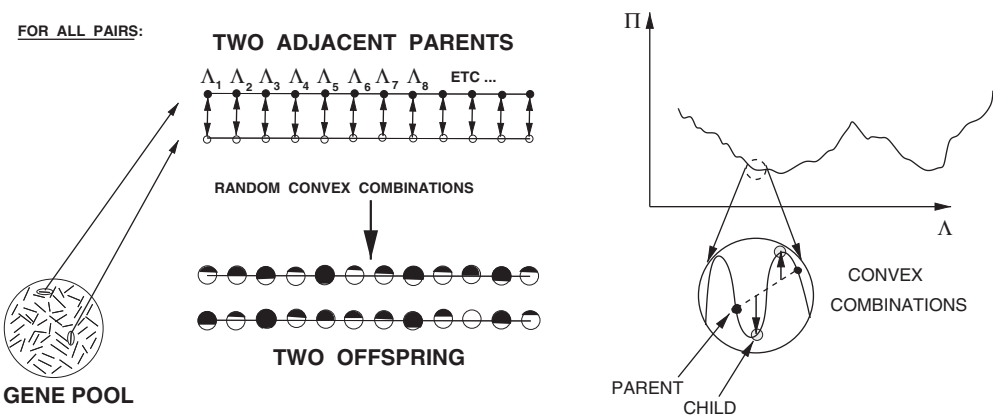

Figure 2. Top: Genetic string combinations. Bottom: A loss of superior older genetic strings if the top parents are not retained.

at random within the design space, are added to the population. Not retaining the parents is suboptimal due to the possibility that inferior offspring will replace superior parents. In addition, parent retention is computationally less expensive, since these designs do not have to be reevaluated in the next generation.

Remark II

Typically, for swarms with a finite number of particles, there will be slight variations in the performance for different random starting configurations. In order to stabilize the objective function's value with respect to the randomness of the swarm starting configuration, for a fixed design ($\mathbf{\Lambda}$ characterized by $\boldsymbol{\alpha}$s and $\boldsymbol{\beta}$s), a regularization procedure is applied within Box 1, whereby the performances of a series of different random starting configurations are averaged until the (ensemble) average converges, i.e. until the following condition is met ($i = 1, 2, \ldots, E$):

$$\left| \frac{1}{E+1} \sum_{i=1}^{E+1} \Pi^{(i)}(\mathbf{\Lambda}^I) - \frac{1}{E} \sum_{i=1}^{E} \Pi^{(i)}(\mathbf{\Lambda}^I) \right| \leqslant \mathrm{TOL} \left| \frac{1}{E+1} \sum_{i=1}^{E+1} \Pi^{(i)}(\mathbf{\Lambda}^I) \right| \tag{15}$$

The index i indicates a starting random configuration that has been generated and E indicates the total number of configurations tested. In order to implement this in Box 1, in STEP 2, one simply replaces 'COMPUTE' with 'ENSEMBLE-COMPUTE', which requires a further inner loop to test the performance of multiple starting configurations. Similar ideas have been applied to randomly dispersed particulate solids in Reference [30].

Remark III

The strategy in Box 1 refines the step sizes to control the iterative error, where the temporal discretization accuracy dictates the upper limits on the time step size ($\Delta t^{\mathrm{DISCRETE}}$).

Remark IV

The strategy in Box 1 can be enhanced several ways. For example, every few generations, the search domain can be restricted and rescaled to be centred around the best current design.

5. NUMERICAL EXPERIMENTS: DESIGN OF A SWARM

We consider the situation illustrated in Figure 3. The components of the initial position vectors of the non-intersecting swarm members, each assigned a mass of 10 kg, were given random values of $-1 \leqslant r_{ix}, r_{iy}, r_{iz} \leqslant 1$. The location of the target was $(10, 0, 0)$. The location of the centre of the (rectangular) obstacle array was $(5, 0, 0)$. A nine-obstacle 'fence' was set up as follows: $(5, 0, 0)$, $(5, 2, 2)$, $(5, 2, -2)$, $(5, 2, 0)$, $(5, 0, 2)$, $(5, -2, -2)$, $(5, -2, 2)$, $(5, -2, 0)$, $(5, 0, -2)$. The radius of 'destruction' for the swarm member-obstacle pair was set to $R = 0.5$. In order to study the effects of the swarm size on the optimal performance, we considered swarms of successively larger sizes, containing $N = 8$, 16, 32, 64 and 128 members. The search space was $10^{-6} \leqslant \boldsymbol{\alpha} \leqslant 10^{6}$ and $10^{-6} \leqslant \boldsymbol{\beta} \leqslant 1$. The number of genetic strings was set to $S = 20$, for $G = 20$ generations keeping the top six offspring of the top six parents. Therefore, after each generation, eight new genetic strings are introduced. Figure 4 depicts the results. The total number of function evaluations of Π is $S + (G - 1) \times (S - Q) = 286$, where $G = 20$ is the number of generations, $S = 20$ is the total number of genetic strings in the population and $Q = 6$ is the number of parents kept after each generation. The total time was set to be one second ($\mathscr{T} = 1$).

From Tables I–IV, there appears to be no convergence in the optima with respect to the number of swarm members. A clear result is that one cannot expect optima for one swarm

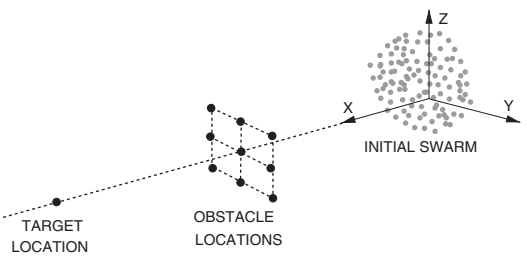

Figure 3. The initial set-up for a swarm example.

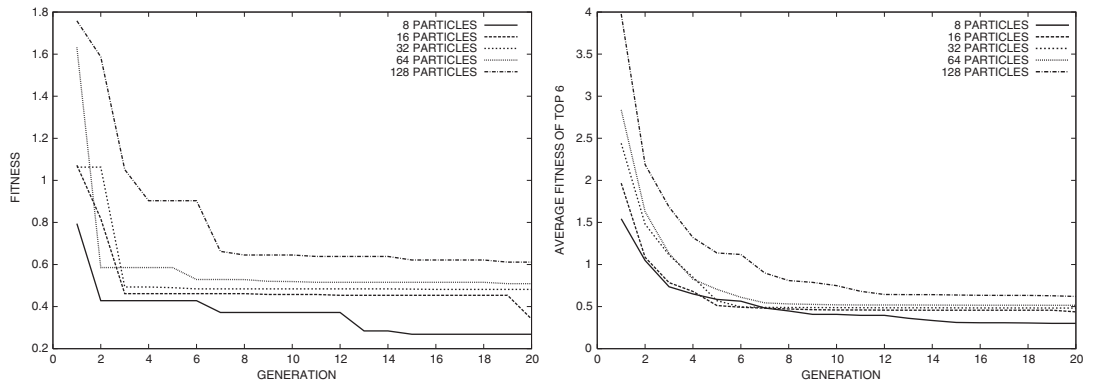

Figure 4. Generational values of (Top) the best design's objective function and (Bottom) the average of the best six designs' objective functions for various swarm member sizes.

Table I. The top fitness and average of the top six fitnesses for various swarm sizes.

Swarm members	Total strings tested	$\frac{\text{Strings}}{\text{Design}}$	Best Π	$\frac{1}{6}\sum_{i=1}^{6}\Pi^{(i)}$
8	1573	5.50000	0.26849	0.30083
16	1646	5.75524	0.34071	0.43753
32	1022	3.57342	0.48168	0.48292
64	1241	4.33916	0.50922	0.51535
128	1970	6.88811	0.61159	0.62105

Table II. The optimal coefficients of attraction and repulsion for various swarm sizes.

Swarm members	α_1^{M-M}	α_2^{M-M}	α^{M-T}	α^{M-O}
8	451470.44867	270188.87859	735534.64355	141859.99366
16	128497.49588	279918.51357	778117.81680	80526.85727
32	111642.28406	564292.53581	872627.48473	7899.69642
64	394344.61827	625999.39911	910734.12027	23961.73083
128	767084.35070	264380.23462	574909.53655	159249.40072

Table III. The optimal exponents of attraction and repulsion for various swarm sizes.

Swarm members	β_1^{M-M}	β_2^{M-M}	β^{M-T}	β^{M-O}
8	0.85555	0.26867	0.43660	0.64331
16	0.17931	0.15648	0.81012	0.83860
32	0.41010	0.04042	0.79951	0.56327
64	0.40304	0.11487	0.74228	0.49764
128	0.59134	0.07889	0.57296	0.83137

size to be optimal for another. In other words, there is no apparent scaling law. In Figures 5–8, frames are shown for approximately every 0.001 s for the 128 particle swarm. The 128 particle swarm bunches up and moves through the obstacle fence unharmed (centred at $(5,0,0)$), by going underneath the central obstacle and between adjacent obstacles. The swarm then unpacks itself, overshoots the target at $(10,0,0)$ and then undershoots it slightly. The swarm starts to home in on the target and concentrate itself at $(10,0,0)$. It is interesting to note that the ratios of optimal member-member repulsion to attraction (ρ^{M-M}) are quite small for the 128 particle swarm, however for other swarm sizes, such as 16 and 32, the optima are relatively large. This implies that bunching up is not necessarily the best strategy to surround the target for every swarm size.

Table IV. The ratios of optimal repulsion and attraction for various swarm sizes.

Swarm members	ρ^{M-M}
8	0.63339
16	10.16223
32	36.46856
64	2.44070
128	0.20406

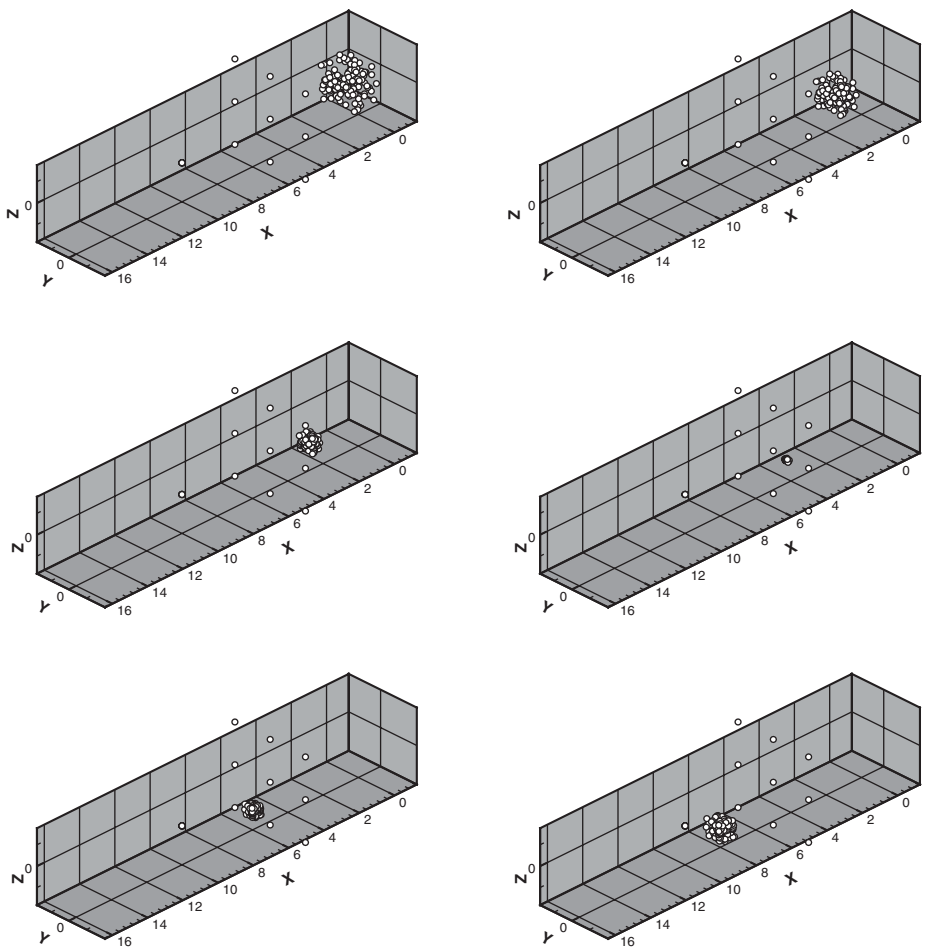

Figure 5. Starting from the top, left to right: Frames 1–6. The swarm (128 swarm members) bunches up and moves through the obstacle fence, under the centre obstacle, unharmed (centred at (5,0,0)), then unpacks itself.

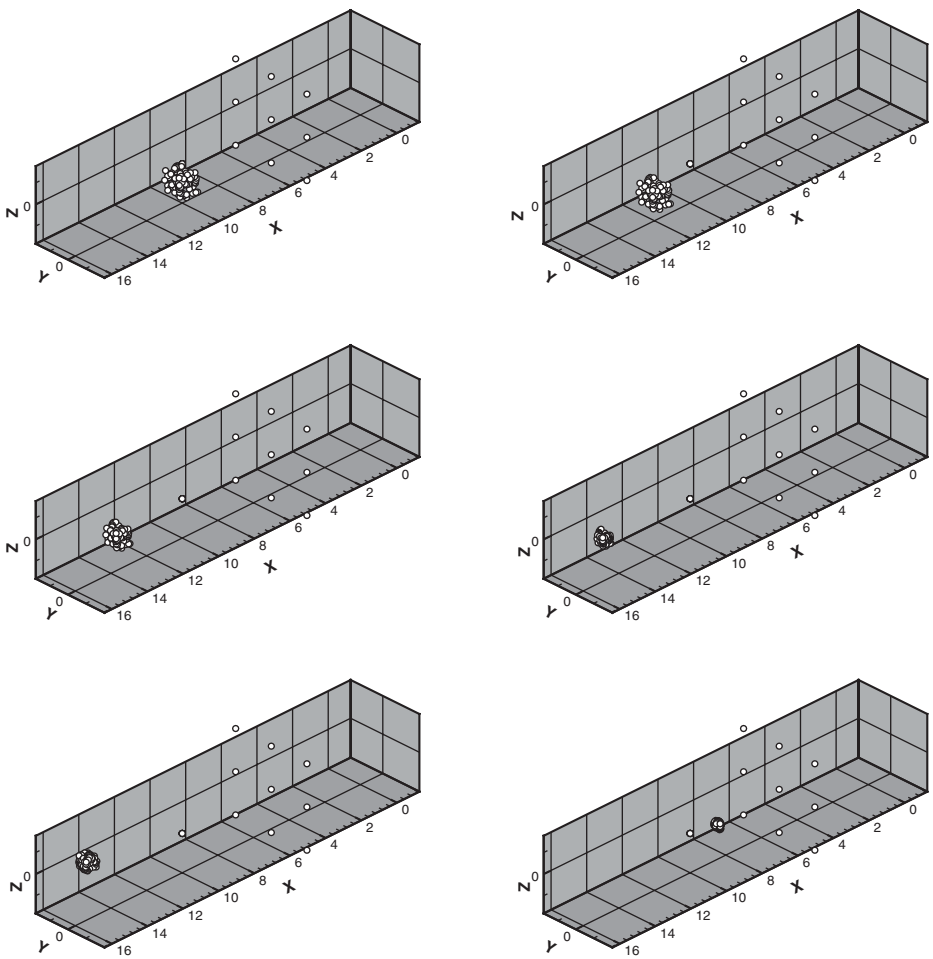

Figure 6. Starting from the top, left to right: Frames 7–10, 15, 20. The swarm then goes through and slightly overshoots the target $(10, 0, 0)$, then undershoots it slightly and starts to concentrate itself.

6. CONCLUSIONS

In many applications, the computed positions, velocities and accelerations of the members of a swarm, for example people or vehicles, must be translated into realizable movement. Furthermore, the communication latency and information exchange is a technical issue that poses a significant hurdle. In practice, further sophistication, i.e. constraints on movement and communication, must be embedded into the computational model for the application at hand. However, the fundamental computational philosophy and modelling strategy should remain relatively unchanged. It is important to remark on a fundamental result found in References [6–9], namely, that if the interaction is only with the nearest neighbours, and if there is no inertial reference point for the swarm members to refer to, instabilities (collisions) may

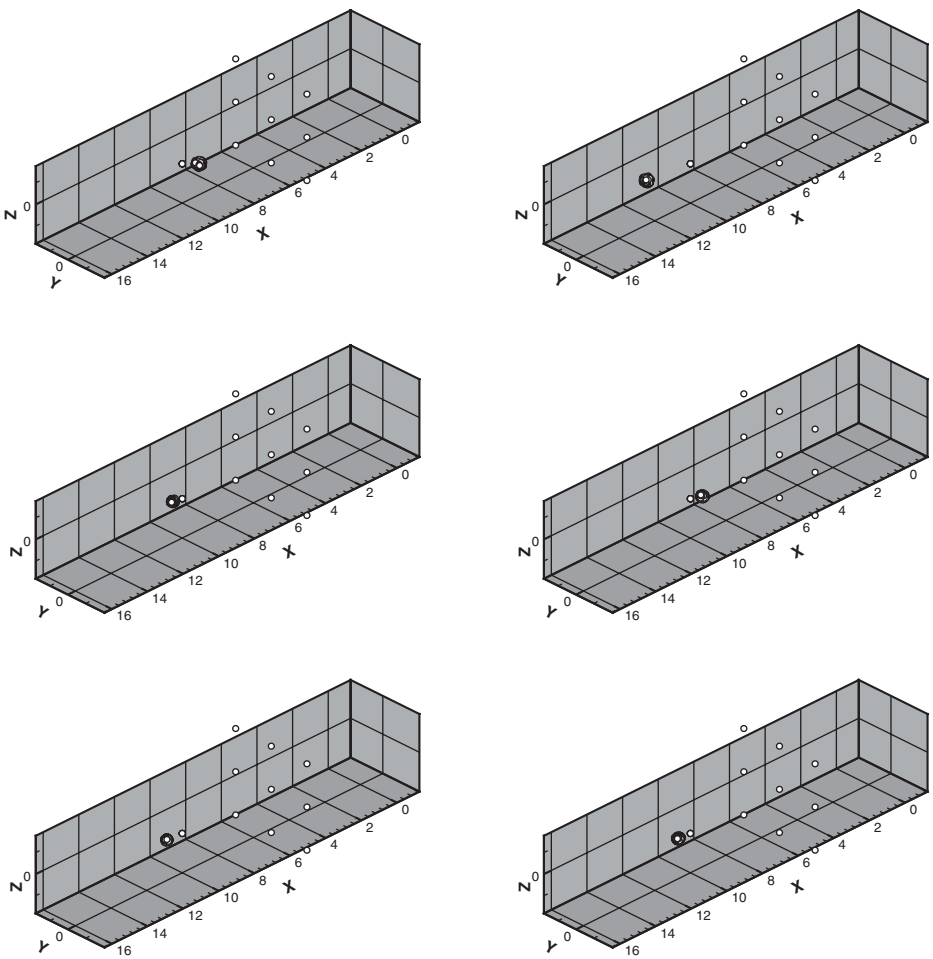

Figure 7. Starting from the top, left to right: Frames 25, 30, 35, 40, 45 and 50. The swarm starts oscillates slightly around the target and then begins to home in on the target and concentrate itself at $(10, 0, 0)$.

occur. In the present analysis, such inertial reference points were furnished by the fact that the members of the swarm knew the absolute locations of the obstacles and target. Also, because the communication for a given swarm member was with all other members, the stability appears to be a non-issue. Furthermore, due to the presence of a $1/r$-type interaction force between the initially non-overlapping swarm members, the centres could not intersect (a singular repulsion term). However, if the target and obstacles begin to move in response to the swarm, which may be the case in certain applications, and the communication between swarm members is only with the nearest neighbours (a possible technological restriction), then instabilities can become a primary concern. Finally, we remark that the specific structure of the interaction forces chosen in this work is only one of many possibilities, and that a different choice of a base interaction force form may allow for even further optimization

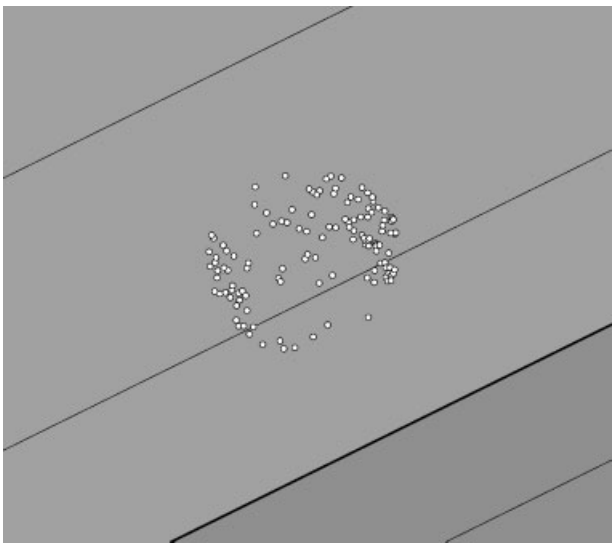

Figure 8. Zoom at frame 50. The swarm surrounds the target at $(10, 0, 0)$.

of the swarm behaviour. For an extensive survey of interaction forces in the closely related field of molecular dynamics, which includes comparisons of the theoretical and computational advantages of each interaction law, we refer the reader to Frenklach and Carmer [31]. The use of other potentials and stability analyses are under current investigation by the author.

REFERENCES

1. Breder CM. Equations descriptive of fish schools and other animal aggregations. *Ecology* 1954; **35**(3):361–370.
2. Gazi V, Passino KM. Stability analysis of swarms. *Proceedings of the American Control Conference*. Anchorage, AK May 8–10, 2002.
3. Liu Y, Passino KM, Polycarpou M. Stability analysis of one-dimensional asynchronous swarws. In *Proceedings of the American Control Conference*, June 2001; 716–721.
4. Liu Y, Passino KM, Polycarpou M. Stability analysis of one-dimensional mobile asynchronous swarms. In *Proceedings of the Conference of Decision and Control*, December 2001; 1077–1082.
5. Bender J, Fenton R. On the flow capacity of automated highways. *Transport Science* 1970; **4**:52–63.
6. Hedrick JK, Swaroop D. Dynamic coupling in vehicles under automatic control. Supplement to *Vehicle System Dynamics* 1994; **23**:209–220.
7. Hedrick JK, Tomizuka M, Varaiya P. Control issues in automated highway systems. *IEEE Control Systems Magazine* 1994; **14**(6):21–32.
8. Swaroop D, Hedrick JK. String stability of interconnected systems. *IEEE Transactions on Automatic Control* 1996; **41**(4):349–356.
9. Swaroop D, Hedrick JK. Constant spacing strategies for platooning in automated highway systems. *Journal of Dynamic Systems, Measurement, and Control* (ASME) 1999; **121**:462–470.
10. Shamma JS. A connection between structured uncertainty and decentralized control of spatially invariant systems. *Proceedings of the 2001 American Control Conference*, Arlington, VA, vol. 4, 2001; 3117–3121.
11. Kennedy J, Eberhart R. *Swarm Intelligence*. Morgan Kaufmann: Los Altos, CA, 2001.
12. Bonabeau H, Dorigo M, Theraulaz G. *Swarm Intelligence*: *From Natural to Artificial Systems*. Oxford University Press: New York, 1999.
13. Zohdi TI. An adaptive-recursive staggering strategy for simulating multifield coupled processes in microheterogeneous solids. *International Journal for Numerical Methods in Engineering* 2002; **53**:1511–1532.
14. Zohdi TI. Large-scale statistical inverse computation of inelastic accretion in transient granular flows. *International Journal of Non-linear Mechanics* 2002; **38**(8):1205–1219.

15. Kokubo E, Ida S. Formation of protoplanets from planetesimals in the solar nebula. *Icarus* 2000; **143**(1): 15–270.
16. Kokubo E, Ida S. On runaway growth of planetesimals. *Icarus* 1996; **123**(1):180–191.
17. Grazier KR, Newman WI, Kaula WM, Hyman JM. Dynamical evolution of planetesimals in the outer solar system I. The Jupiter/Saturn zone. *Icarus* 1999; **140**(2):341–352.
18. Grazier KR, Newman WI, Varadi F, Kaula WM, Hyman JM. Dynamical evolution of planetesimals in the outer solar system II. The Saturn/Uranus and Uranus/Neptune zones. *Icarus* 1999; **140**(2):353–368.
19. Supulver KD, Lin DNC. Formation of icy planetesimals in a turbulent solar nebula. *Icarus* 2000; **146**(2): 525–540.
20. Tanga P, Babiano A, Dubrulle B, Provenzale A. Forming planetesimals in vortices. *Icarus* 1996; **121**(1): 158–170.
21. Weidenschilling SJ, Spaute D, Davis DR, Marzari FR, Ohtsuki K. Accretional evolution of a planetesimal swarm. *Icarus* 1997; **128**(2):429–455.
22. Barranco J, Marcus P, Umurhan O. Scaling and asymptotics of coherent vortices in protoplanetary disks. In *Studying Turbulence Using Numerical Simulation. Databases-VIII*, *Proceedings of the Summer Program*, Center for Turbulence Research. Stanford University Press: Stanford, 2001.
23. Barranco J, Marcus P. Vortices in protoplanetary disks and the formation of planetesimals. In *Studying Turbulence Using Numerical Simulation. Databases-VIII*, *Proceedings of the Summer Program*, Center for Turbulence Research. Stanford University Press: Stanford, 2001.
24. Axelsson O. *Iterative Solution Methods*. Cambridge University Press: Cambridge, 1994.
25. Ostrowski A. *Solution of Equations and Systems of Equations*. Academic Press: New York, 1966.
26. Ortega J, Rockoff M. Nonlinear difference equations and Gauss-Seidel type iterative methods. *SIAM Journal on Numerical Analysis* 1966; **3**:497–513.
27. Kitchen J. Concerning the convergence of iterates to fixed points. *Studia Mathematica* 1966; **27**:247–249.
28. Ames WF. *Numerical Methods for Partial Differential Equations* (2nd edn). Academic Press: New York, 1977.
29. Davis L. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.
30. Zohdi TI. Computational optimization of vortex manufacturing of advanced materials. *Computer Methods in Applied Mechanics and Engineering* 2001; **190**(46–47):6231–6256.
31. Frenklach M, Carmer CS. Molecular dynamics using combined quantum and empirical forces: application to surface reactions. *Advances in Classical Trajectory Methods* 1999; **4**:27–63.

