Chapter 7
MLC tactics and strategy

"Heavy is the brick of reality on the strawberry cake of our illusions."

- Gilles "Boulet" Roussel, cartoonist, *Translated from its Twitter account @Bouletcorp, December 10th 2013*

If the ideal MLC experiment existed, this chapter would not be needed. The literature contains many examples of control methods that fail the translation from a numerical study to the corresponding experimental demonstration. MLC removes one of the major obstacles: the discrepancy between the employed model and the physical reality of the experiment. Nonetheless there still exist many pitfalls that lead to disappointing results. Often, these results relate to the difference between the idealized experiment and reality. Through our first applications of MLC in experiments we have explored many pitfalls and have developed a mitigation strategy. This chapter guides experimental control demonstrations with MLC. The advice may also apply to numerical studies.

7.1 The ideal flow control experiment

The ideal flow control experiment — or any other control experiment — does not exist. If it existed, it would have the following properties:

Full knowledge about the dynamics: The evolution equation $d\mathbf{a}/dt = \mathbf{F}(\mathbf{a},\mathbf{b})$ and the parameters of the dynamics $\mathbf{F}$, like the Reynolds or Mach number, are exactly known. A corollary is that reproducibility is guaranteed.

Accurate measurements: The measurement equation $\mathbf{s} = \mathbf{G}(\mathbf{a},\mathbf{b})$ is known with perfect precision and the signals $\mathbf{s}(t)$ are accurately recorded.

Ideal cost function: The cost function $J(\mathbf{a},\mathbf{b})$ describes the quantity which shall be optimized and which can be measured in the experiment.

Known noise: Any noise terms affecting the dynamics, the measurement equation or the cost function are accurately modeled and accounted for. A corollary is the reproducibility of arbitrary ensemble averages.

Controllability: Any desired final state **a** of the system can be reached with finite actuation in finite time. At a minimum, the achieved change of cost function due to MLC is so impressive, that the results merit a career promotion and convince the funding agencies to invest more money.

Infinitely powerful computer: The control law $\mathbf{b} = \mathbf{K}(\mathbf{s})$ is computed instantaneously. At minimum, the actuation command is computed with predictable small time delay.

No aging: Of course, the ideal experiment never breaks, nor suffers slowly drifting external parameters such as atmospheric pressure and temperature, nor is effected by opening and closing doors of the room, nor experiences any changes which are not reflected in the dynamics **F**, in the measurement function **G** and in the cost function J.

Infinite resources: The perfect experiment can be operated until the cost function value is converged and until MLC is converged to the globally optimal control law.

Arguably, the ideal flow control experiment is an instance of a direct Navier-Stokes simulation on an infinitely powerful computer. When building a MLC control experiment the best degree of perfection shall be reached with given resources. In the following sections, we address different aspects that one needs to keep in mind when preparing an MLC experiment. Some decisions can be made to prepare a perfect experiment. And other decisions deal with imperfections of an existing experiment. This chapter can be considered as a checklist of important aspects that need to be taken into account in order to maximize the chances of success and the quality of the results.

7.2 Desiderata of the control problem — from the definition to hardware choices

In this section, we outline desiderata of the multi-input multi-output (MIMO) control problem. This comprises the definition of the cost function (Sec. 7.2.1), the choice of the actuators (Sec. 7.2.2), and the choice of the sensors (Sec. 7.2.3), including a pre-conditioning of the actuation command and sensor signals for MLC. In Section 7.2.4) we remind the reader that the search space for control laws is a design choice as well. Many aspects are trivial. This does, however, not exclude that one or the other aspect might be overlooked. Even the most sophisticated control problem can be decomposed in myriad of trivialities.

7.2.1 Cost function

The purpose of any actuation is to change the behavior of the plant to be more desirable. The corresponding performance measure is generally called the *cost function* in control theory and is generally a positive quantity to be minimized. A control problem may have a 'natural' cost function comprising the departure of the system from the ideal state J_a and the associated cost J_b. We take the aerodynamic drag reduction of a car by unsteady actuators as an example. The propulsion power to overcome the drag F_D at velocity U reads $J_a = UF_D$ and is ideally zero. The actuation power by blowing can be estimated by the energy flux $J_b = \rho S_b \overline{U_b^3}/2$, where ρ is the density of the fluid, S_b is the area of the actuator slot, and $\overline{U_b^3}$ the third power of the actuator velocity averaged over the slot area and time. The resulting cost function reads

$$J = \underbrace{F_D U}_{J_a} + \underbrace{\rho S_b \overline{U_b^3}}_{J_b}. \tag{7.1}$$

Let J_u be the parasitic drag power without forcing. Then, the net energy saving reads $\Delta J = J_a + J_b - J_u$ and shall be maximized, or, equivalently the total energy expenditure $J_a + J_b$ minimized. The ratio between energy saving and actuation energy $(J_u - J_a)/J_b$ defines the efficiency of the actuators and should be much larger than unity. We shall not pause to consider potential improvements on this particular cost function.

Two common cases may obstruct the definition of a 'natural' cost function. First, the cost function cannot be measured in experiments. For instance, the test section of the wind tunnel may not have a scale and the drag force cannot be measured. In this case, a surrogate cost function needs to be defined. A good surrogate function is expected to follow approximately — at least qualitatively — the changes of the intended quantity for all considered control laws. In case of the car example, the base pressure coefficient is a possible choice.

A second challenge may be the incomparableness of control goal J_a and actuation cost J_b. This may, for instance, be manifested in different physical units of J_a and J_b. The control goal may be to improve mixing as quantified by an non-dimensional entropy and the actuation cost may be measured in Watts. In this case, cost function decomposition in a state and actuation contribution $J = J_a + J_b$ is even dimensionally wrong and there is no a priori performance benefit $J_{a,u} - J_a$ which is worth J_b in actuation energy. In this case, the actuation cost is typically rescaled with a penalization parameter γ,

$$J = J_a + \gamma J_b. \tag{7.2}$$

In the following, we assume that all quantities are non-dimensionalized and $\gamma > 0$ is a positive real number. Let us further assume that the experimentalist decides that the performance benefit $J_1 = J_u - J_a$ is worth the actuation cost J_2. In this case, the corresponding penalization parameter reads $\gamma = J_1/J_2$. The reference values J_1 and J_2 might be obtained from another control experiment. For instance J_1 may be the

performance benefit from the actuation cost J_2 from periodic forcing. In this case, the same choice of γ leads to a lower J-value (7.2) for MLC if the actuation benefit $J_u - J_a$ is larger at the same actuation cost or if the same actuation benefit is achieved at lower actuation cost.

7.2.2 Actuators

The goal of actuation is to modify the system behavior to reduce the cost function. This self-evident goal may not be achieved by the actuators in real-world experiments, as the authors have witnessed a number of times.

The goal may be to reduce jet noise with plasma actuators at the jet nozzle exit. But these actuators may be orders of magnitude more noisy than the jet. Then, the best control strategy is to keep actuation off. The goal may be to reduce the drag of a bluff body, but the blowing actuators increase drag for all tested periodic forcing. The goal may be to mitigate flutter in a turbine, but the zero-net-mass-flux actuator in front the propeller has no perceivable effect. The goal may be to reduce skin friction by mitigating incoming Tollmien-Schlichting waves, but the actuator only triggers three-dimensional transition, which is far worse for drag. An ill-chosen actuation strategy can undermine the entire control campaign. Hence, it is very comforting if there exists a control study in the considered plant for which the actuation has been shown to have a beneficial affect on the cost function.

A second, much easier aspect to consider is the formulation of the regression problem for MLC. As the constants for genetic programming are chosen from a parameter range, say $[-5,5]$, the actuation command $\mathbf{b}$ and sensor signal $\mathbf{s}$ should also be of order one. Evidently, the learning of any regression technique is accelerated, if it has to find mappings between order-one quantities as opposed to translating a $O(10^6)$ sensor signal into $1+O(10^{-6})$ actuation command. In the simple case, that actuation may only be 'on' or 'off', as with some valves, and a good choice of the actuation command is 0 for 'off' and 1 for 'on'.

7.2.3 Sensors

The purpose of the sensors is to infer control-relevant features of the flow state. In linear control theory, the sensors ideally allow one to estimate the controllable subspace of the system. In a turbulent flow, the choice of the type, the locations, and the dynamic bandwidths of the sensors should reflect the assumed enabling actuation mechanism. We refer to [43] and references therein for corresponding guidelines.

Secondly, the sensor signals are ideally hardly correlated so that each sensor provides new information about the state.

Thirdly, we reiterate that the sensor signals should be suitably centered and scaled to become of order unity. Let s_i^* be the raw ith sensor signal. Let $s_{i,\min}^*$ and $s_{i,\max}^*$ be the minimal and maximal values, respectively. Then, learning of MLC is accelerated by using the normalized signal,

$$s_i = \frac{s_i^* - s_{i,min}^*}{s_{i,max}^* - s_{i,min}^*}, \tag{7.3}$$

as input in the control law.

As discussed in the examples of Chapter 6, fluctuations may be a better input for the control law than the raw signal which is affected by drifts. Frequency filtering of the raw signals may also improve performance of MLC. Yet, we strongly disadvise against any of such filtering — at least in the first applications of MLC. Any filtering is an expression of an a priori assumption about the enabling actuation mechanism. This bias may prevent MLC to find a more efficient unexpected control, as we have witnessed in half of our studies.

7.2.4 Search space for control laws

The search space for control laws is a potentially important design parameter of MLC. In Chapter 3, the linear-quadratic regulator $\mathbf{b} = \mathbf{K}\mathbf{a}$ was shown to be optimal for full-state feedback stabilizing a linear plant. This result has inspired the MLC search for full-state, autonomous, nonlinear feedback law $\mathbf{b} = \mathbf{K}(\mathbf{a})$ stabilizing the nonlinear dynamical system of Chapter 5. Similarly, MLC has improved the performance of the experiments in Chapter 6 with a sensor-based, autonomous, nonlinear feedback law $\mathbf{b} = \mathbf{K}(\mathbf{s})$. However, the optimal control of a linear plant which perturbed by noise involves filters of the sensor signals, as discussed in Chapter 4.

For the simple nonlinear plant in Chapter 5, we have made the case that closed-loop control is not necessarily more efficient than open-loop control. The generalized mean-field system of this chapter can easily be modified to make periodic forcing more optimal than full-state autonomous feedback. In some experiments, a periodic high-frequency forcing $b^*(t) = B^* \cos \omega^* t$ may stabilize vortex shedding much better than any sensor-based autonomous feedback. In this case, any deviation of the actuation command from a pure periodic clockwork might excite undesirable vortex pairing. A simple solution for the MLC strategy is to consider the optimal periodic forcing as an additional 'artificial' sensor $s_{N_s} = b^*$ and, correspondingly, as input in the feedback law $\mathbf{b} = \mathbf{K}(\mathbf{s})$. Now, this law depends explicitly on time and is not autonomous anymore. The plant sensors may help to adjust the forcing amplitude, like in extremum seeking control, or introduce other frequencies. The authors have supervised MLC experiments in which this approach has lead to pure harmonic actuation, to multi-frequency forcing and to nonperiodic forcing as best actuation. The search space for optimal control laws may also include multi-frequency inputs and filters. Our preferred strategy is to start with direct sensor feedback $\mathbf{b} = \mathbf{K}(\mathbf{s})$

and integrate the best periodic periodic forcing as control input before exploring more complex laws.

7.3 Time scales of MLC

The MLC experiment is affected by three time scales. First, the inner loop has a time delay from sensor signals to actuation commands. This time scale is dictated by the employed hardware, for instance, the computating time for the control law (Sec. 7.3.1). Second, the inner loop has a plant-specific response time, i.e. a time delay from the actuation to the sensed actuation response. This time scale is related to the plant behavior, e.g. the choice of the actuators and sensors and the flow properties (Sec. 7.3.2). Third, the outer MLC loop has a learning time governed by the number and duration of the cost function evaluations before convergence (Sec. 7.3.3). In this section, we provide recommendations how to deal with all three time scales.

7.3.1 Controller

In the computational control studies of Chapters 2, 4 and 5, we write the control law as

$$\mathbf{b}(t) = \mathbf{K}(\mathbf{s}(t)), \tag{7.4}$$

i.e. the actuation reacts immediately on the sensor signals. In a real-world experiment, there is a time delay τ_b from sensing to actuation leading to another optimal control law,

$$\mathbf{b}(t) = \mathbf{K}_{\tau_b}(\mathbf{s}(t-\tau_b)). \tag{7.5}$$

This time scale comprises three delays: (1) the propagation time from the sensor location to the computational unit, e.g. the propagation of the pressure from the body surface to the measurement device in tubes; (2) the computational time needed to compute $\mathbf{K}_{\tau_b}(\mathbf{s})$; and (3) the time from the computed actuation command $\mathbf{b}$ to the actuator action. The times (1) and (3) would not occur in a computational study. One could argue that (1) and (3) belong to the physical response time of the plant or to the controller. In this section, we do something much simpler: We focus on the computational time and ignore the plant-related delays. In addition, we assume that τ_b is fixed and small with respect to the time scale of the plant dynamics, $\tau_b \ll \tau$. The plant time scale might be quantified by or related to the first zero of the sensor-based correlation function,

$$C_{\mathbf{s}}(\tau) = \overline{\mathbf{s}(t)\cdot\mathbf{s}(t-\tau)} = 0. \tag{7.6}$$

In fact, we can be less restrictive with respect to τ_b if the sensor signal at time $t-\tau_b$ allows a good prediction of the values at time t,

$$\mathbf{s}(t) = \phi_{\tau_b}\left[\mathbf{s}(t-\tau_b)\right]. \tag{7.7}$$

Here, ϕ_{τ_b} is the propagation map which may also be identified with the discussed regression techniques. This can, for instance, be expected for oscillatory dynamics. In this case, the instantaneous control law (7.4) can be recovered from Eq. (7.5) using the predictor (7.7).

In the sequel, τ_b will be considered as the computational time. This can easily be estimated for a binary tree of genetic programming with single input ($N_b = 1$). Let us assume that the tree is maximally dense and has the depth 10. In this case, the evaluation requires $2^9 + 2^8 + \ldots + 1 = 1023$ operations. A cheap modern laptop has roughly 1 GFlop computing power, i.e. can compute 10^9 operations per second. Hence, the computation time for one actuation command is 1 μs, or one millionth of a second. This is one order of magnitude faster than an ultra-fast sampling frequency 100 kHz of modern measurement equipment. Typical time scales in wind tunnels range from 10 to 100 Hz. Thus, $\tau_b/\tau \leq 10^{-4}$ and the computation time τ_b can be considered small if not tiny by a very comfortable margin. The computer in the TUCOROM experiment (Sec. 6.3) has 1 TFlop computing power, which would add another factor of 1000 as safety margin. The computational time for the control law is apparently not a major concern.

The real challenge is transferring the LISP expression of the control law to the computational unit. The LISP expression may be pre-compiled and transferred to the computational unit before evaluation of the cost function. In this case, the computation time is minimal. The LISP expression may also be read from the hard disk every computation cycle and interpreted by the computational unit. The corresponding time is orders of magnitude slower.

In other words, the potential bottleneck is the communication between the inner RT loop and the outer learning loop. Many communication protocols and architectures exist.[1] We consider here two philosophies, the mentioned interpreter implementation (LISP parser) and the pre-compiled solution.

RT LISP parser: This parser function takes the LISP expression and sensor readings as the inputs and outputs the control commands. The principle for this function is simple but auto-regressive which means that the CPU and memory cost is bounded by $2^l - 1$, where l represents the depth of the expression tree, i.e. function complexity. Due to CPU and memory usage it can lead to low RT scheduler frequency and overruns. This strategy is advisable only with Concurrent, DSpace or equivalent high-end RT systems. Functions of this type have already been written and tested in Simulink®, Labview®, Fortran, Matlab® and C++. This approach has been employed in the TUCOROM mixing layer experiment (see Sec. 6.3).

[1] In principle, the file exchange can happen over different locations in a cloud, with a fast computational unit close to the experiment and the MLC learning performed remotely.

Pre-compiled RT controller: In this case, MLC generates the code for the controller and compiles it. The code will contain the whole generation to be evaluated. The generated code will also make the update of the control law after each evaluation and account for the transient time between evaluations. The cost function values may be returned to the outer MLC loop either after each evaluation or after grading the whole generation. The second communication protocol is evidently much faster than the RT LISP parser and advised for slower systems, like the Arduino processors of Sec. 6.2.

The choice of the communication depends on the kind of RT system available: a parser allows one to update the control law on the fly but is more demanding in memory and CPU resources. This strongly indicates the use of a proper RT dedicated computer with an efficient scheduler. On the other hand, a compiled microcontroller is less flexible: the change of control law is either a scheduled event or enforces a new compilation. The pre-compiled controller is not demanding in terms of programming skills and can be operated with low-cost hardware.

The code of the RT system and of MLC may be written different languages. The authors have successfully implemented the Matlab® code `OpenMLC` on:

- Matlab® and Simulink® (for ODE integration, as in Chapters 4 and 5);
- C++, (Arduino) as in Sec. 6.2;
- Fortran (CFD code) and
- Labview® as in Sec. 6.3.

The use of any other language should not be more than a 3 hour exercise for a skilled programmer.

7.3.2 Response time of the plant

Fluid flows are convective in nature. This leads to a time delay between actuation and sensing. In feedforward control, actuators respond to oncoming perturbations detected by upstream sensors. The mitigation of Tollmien-Schlichting waves in laminar flows is one example for which this approach is shown to be successful. In a feedback arrangement, the sensors record the actuation effect with a time delay. This time delay can often be estimated by the convection time. Most turbulent flows fall in this category, in particular drag reduction or mixing enhancement in bluff-body wakes.

In model-based control design, time delays are known to cause undesirable control instabilities and lack of robustness. For model-free control, the time delay between actuation and sensing is not a problem per se. Several MLC-based actuation strategies have successfully worked with a time delay of 1 to 2 characteristic periods. Yet, a degrading control performance may be caused by the increasing uncertainty of the flow state between actuation and sensing. Ideally the time delay between actuation and sensing should be long enough so that the actuation effect is clearly identifiable beyond the noise threshold but short enough so that the state uncertainty

is still acceptable. This aspect is beautifully demonstrated in wake stabilization experiment by Roussopoulos [226].

7.3.3 Learning time for MLC

The learning time T_{MLC} of MLC using genetic programming with N_g generations and N_i individuals reads:

$$T_{\mathrm{MLC}} = N_g \times N_i \times (T_{\mathrm{transient}} + T_{\mathrm{evaluation}}) \tag{7.8}$$

where $T_{\mathrm{transient}}$ is the transient time for a new control law and $T_{\mathrm{evaluation}}$ the evaluation time of the cost function.

A proper planning of a MLC experiment includes good choices for the parameters that determine T_{MLC} in Eq. (7.8). A minimal transient time $T_{\mathrm{transient}}$ is determined by the flow physics. The time for a transient from an actuated to an unforced state (or the inverse process) is a good reference value. The evaluation time $T_{\mathrm{evaluation}}$ should be large enough so that the *approximate* ordering of the J-values is right. This requirement is distinctly different from requiring a 1% or 5% accuracy of the J-value. The evaluation time for MLC can be expected to be much shorter if high accuracy is not required. One might want to invest in the evaluation time during the final generations when the explorative phase turns into an exploiting phase of MLC.

The next decision concerns the population size N_i and the number of generations N_g. In larger-scale experiments, the measurement time T_{MLC} is expensive and given as a side constraint. Thus, the decision is how to distribute a given number of runs $N_i \times N_g$ among population size and convergence. Given, for instance, $N_i \times N_G = 1000$ evaluations in a measurement campaign, one is left with several possibilities:

1 **generation of** 1000 **individuals.** The search space is explored as widely as possible but no convergence is achieved: this is a Monte-Carlo process.

10 **generations of** 100 **individuals.** The search space is initially less explored, but the evolution will provide increasingly better individuals.

100 **generations of** 10 **individuals.** The search space is hardly explored initially. If the initial generations include winning individuals, the evolution may provide progressively better values, like in gradient search. Otherwise, the evolution may terminate in a suboptimal minimum for the cost function.

In other words, the choice depends on the assumed complexity of the cost function variation or number of local minima. The more complex the landscape of the cost function is, the more MLC will profit from exploration with a large number of individuals in a given generation. The more simple the landscape, the more MLC will benefit from exploitation with large number of generations. A good strategy is to start with a large population at the beginning for good exploration and switch to a much smaller population for good exploitation. The degree of exploration and exploitation is also determined by the MLC parameters as will be elaborated in the following section.

7.4 MLC parameters and convergence

In this section, we discuss the convergence process of MLC based on genetic programming. The discussion is representative for other evolutionary algorithms as well. The purpose is to minimize the MLC testing time (7.8), more precisely $N_i \times N_g$, for an acceptable control law.

Most regression techniques are based on iterations. Gradient search, like Newton's method, can be conceptualized as an iteration of a population with a single individual $N_i = 1$ sliding down a single smooth local minimum. This process is called *exploitation*. Evolutionary algorithms, like genetic programming, assume multiple minima and hence evolve a large number of individuals $N_i \gg 1$. The search for new minima is called *exploration*. Evidently, evolutionary algorithms need to make a compromise between exploring new minima and exploiting found ones.

In the following, we discuss diagnostics of the convergence process (Sec. 7.4.1), MLC parameters setting the balance between exploration and exploitation (Sec. 7.4.2), and pre-evaluation of individuals to avoid testing unpromising candidates (Sec. 7.4.3).

7.4.1 Convergence process and its diagnostics

We analyze a complete *MLC run* from the first to the last generation N_g with N_i individuals each. Focus is placed on the cost function values as performance metrics. Let J_i^j be the cost function value of the ith individual in the jth generation. These values are sorted as outlined in Chapter 2. First, within one generation the cost function values are numbered from best to worst value:

$$J_1^j \leq J_2^j \leq \ldots \leq J_{N_i}^j \quad \text{where} \quad j = 1, \ldots, N_g. \tag{7.9}$$

Second, elitism guarantees that the best individual is monotonically improving with each generation, neglecting noise, uncertainty and other imperfections of the plant:

$$J_1^1 \geq J_1^2 \geq \ldots \geq J_1^{N_g}. \tag{7.10}$$

Figure 7.1 illustrates the J-values of the first regression problem in Sec. 2.3.1. The performance of MLC may be assessed from the following quantities. The best individual J_1^j, $j = 1, \ldots, N_g$ is depicted in the bottom blue curve in Fig. 7.1. MLC appears to be converged after about 30 generations.

The whole population may be characterized by the median value $J^j_{\lfloor N_i/2 \rfloor}$, where the brackets $\lfloor \cdot \rfloor$ represent the nearest integer. The J-values vary over orders of magnitude in each generation so that the mean value or variances are strongly affected by the worst outliers. Arguably, the median value is a much better parameter of the evolution of the whole population. The difference between best and median individual is an indicator of the diversity of the population. The diversity is needed to explore new potentially better minima.

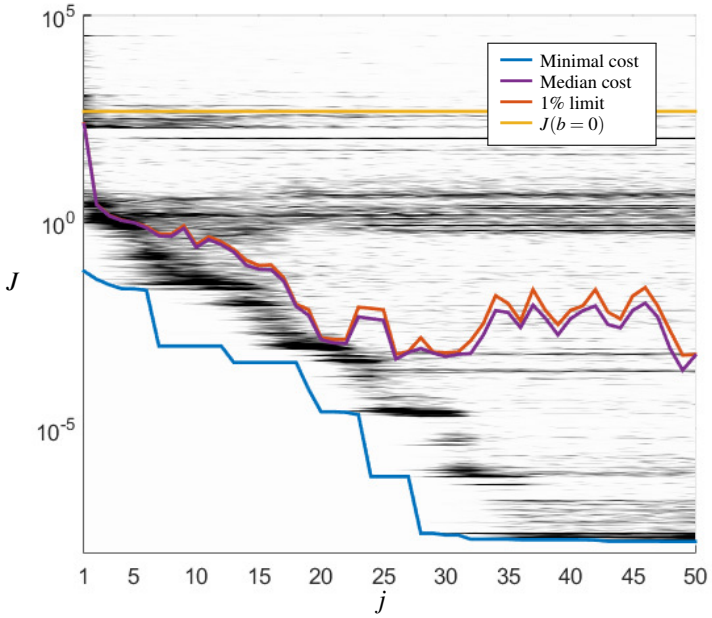

Fig. 7.1 MLC run for example in Sec. 2.3.1 featuring the best and median J-values In addition, the 1% limit indicates that all individuals above this line cumulate a 1% contribution to the genetic content of the next generation. The yellow line indicates the cost of all identically null individuals.

The diversity of the evolutionary process is also characterized by another J-value. The larger the J-value of an individual the less probable is the selection for any genetic operation into the next generation. The J-value above which only 1% of the population is selected for an operation is called the *1% limit* and displayed in Fig. 7.1. This limit follows roughly the median in the presented example.

A peculiar feature of Fig. 7.1 is the yellow curve corresponding to the identically vanishing control function. This *zero-individual* can easily be produced from any individual by multiplying it with 0.

A more detailed insight in the evolution dynamics may be gained from population densities. Let $J_{\min}$ and $J_{\max}$ be the best and worst cost values found in the whole MLC run. We equipartition $\log_{10} J$ in 1000 equidistant intervals $I_k^{\text{pdf}} = \left[J_{k-1}^{\text{pdf}}, J_k^{\text{pdf}}\right]$ where

$$J_{\min} = J_0^{\text{pdf}} < J_1^{\text{pdf}} < \ldots < J_{1000}^{\text{pdf}} = J_{\max}.$$

The logarithmic equipartition separates values which are orders of magnitude different. Let

$$n_k^j := \text{card}\left\{J_i^j : i = 1, \ldots, N_i \wedge J_i^j \in I_k^{\text{pdf}}\right\} \tag{7.11}$$

be the number of J values of the jth generation in the kth interval. Figure 7.2 displays the population density n_k^j/N_i. We see islands of highly populated similar J-values

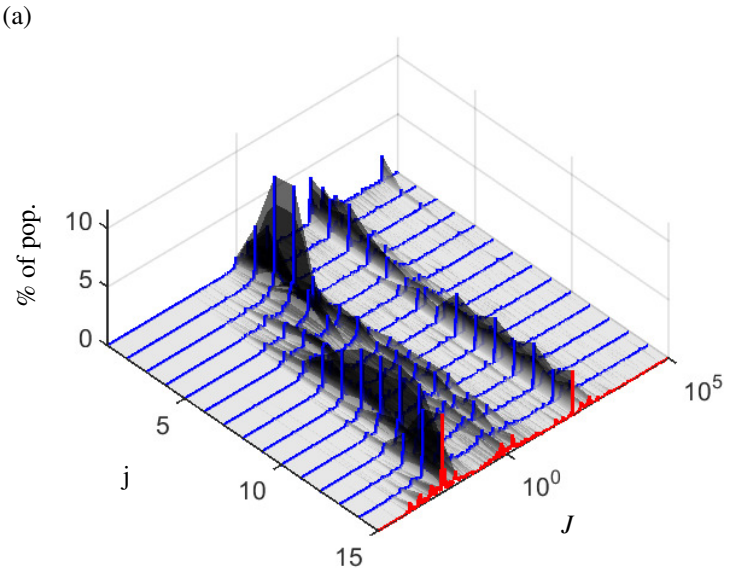

Fig. 7.2 Same MLC run as in Fig. 7.1 but with a 3-dimensional view of the population density (7.11).

over several generations. These islands tend to get initially increasingly populated until better islands are discovered.

The evolution of the generations can be beautifully visualized with multi-dimensional scaling. This leads to a mathematically well-defined illustration of the principle sketch of Fig. 2.12, i.e. indicates the landscape of J spanned by the population of control laws. Let $\mathbf{K}^i(\mathbf{s})$ denote the ith control law of an MLC run, i.e. $i = 1,\ldots,N_i N_g$. Let $\gamma^i = \left(\gamma_1^i, \gamma_2^i\right)$ be a two-dimensional vector corresponding to the control law. Let $D^{ij} = \langle \|\mathbf{K}^i(\mathbf{s}) - \mathbf{K}^j(\mathbf{s})\| \rangle$ be a suitable metric between the ith and jth control law. Kaiser et al. suggest such a metric in [155]. Then multi-dimensional scaling provides a configuration of γ^i which optimally preserves the metric $\|\gamma^i - \gamma^j\| \approx D^{ij}$. Figure 7.3 shows an example of a closed-loop control experiment targeting drag reduction of a car model. The illustration indicates a clustering and scattering of control laws with an apparent minima in the lower left corner. Similar visualizations of other MLC experiments showed far more complex topologies. The understanding of these topologies is still a subject of active research.

7.4.2 Parameters

As outlined in Sec. 7.3.3, a large population size N_i promotes exploration while a long evolution N_g supports convergence. In addition, the learning rate from genera-

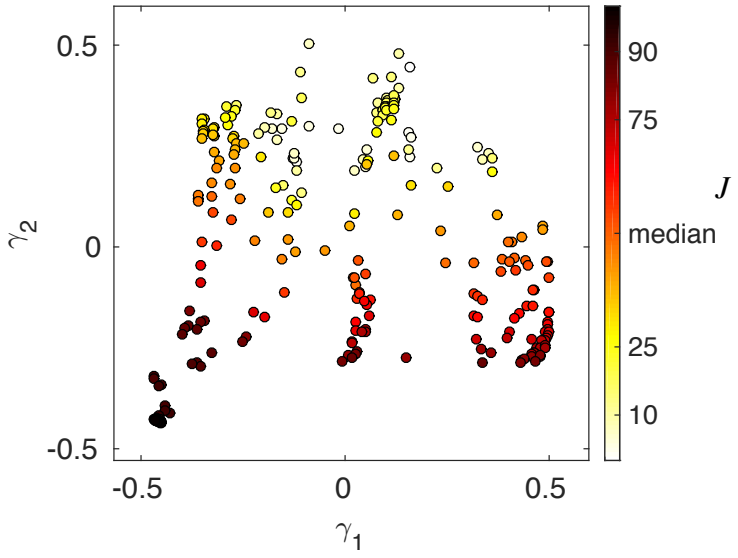

Fig. 7.3 Visualization of an experimental MLC run for drag reduction of a car model with multi-dimensional scaling (Courtesy: Ruiying Li for the data and Eurika Kaiser for the visualization). The circles correspond to all individuals of a converged evolution with $N_i = 50$ and $N_g = 5$. The J-performance of the individuals are color-coded with respect to the ordering. A control law at the 90th percentile for instance illustrates that 90% of the J-values are worse and only 10% of the J-values are better. The plane (γ_1, γ_2) preserves a metric between control laws in an optimal manner (multi-dimensional scaling, see [155] for details).

tion to generation with given population size is affected by several MLC parameters. The visualizations of Sec. 7.4 give a good indication of this learning process.

The learning speed of MLC is influenced by the genetic operation probability and the selection of individuals.

Elitism N_e: A non-vanishing N_e ensures that the best performing individuals reappear in the next generation and are not forgotten. Generally, $N_e = 1$ is a good minimal choice. A larger number tends to copy very similar individuals in the new generation, i.e. neither contributes to diversity nor to performance.

Genetic operation probability P_c, P_m and P_r: The sum of the three probabilities must be unity, implying that only two probabilities can be independently chosen. As discussed in Chapter 2, replication ensures the memory of good individuals, crossover promotes exploitation and thus better individuals, and mutation promotes exploration and diversity of the population. Exploration, i.e. finding the right minima, is initially more important that exploiting the minima that have already been discovered. Good starting values for $N_i \sim 100$ were found to be $(P_r, P_c, P_m) = (0.1, 0.6, 0.3)$. $P_r = 0.1$ appears to be a uniformly good choice independent of the population size N_i. For much larger population size $N_i \gg 100$, a smaller P_m and larger P_c are advisable. Conversely, smaller population sizes profit from choosing a larger mutation rate, e.g. $P_m = 0.4$.

Selection: The arguments of the genetic operations are determined by a stochastic selection process of the individuals. Like genetic operations, the selection may promote exploitation or exploration. The harsher the bias towards best performing individuals, the larger the probability that the local minima will be exploited. The individuals are chosen with equal probability in a tournament type selection. The harshness of the selection is set by the tournament size. Setting $N_p = 7$, enforces that the upper 50% of the current population accounts for the genetic content of 99% of the next generation for $N_i = 100$.

7.4.3 Pre-evaluation

The initial generation is composed of randomly chosen individuals. Evidently, one can expect that most individuals are non-performing, probably performing worse than the zero-individual $b \equiv 0$. If random control laws would improve the cost function, then control design would be simple.

Hence, a pre-evaluation of initial individuals is advised. Zero-individuals, for instance, should be replaced by other random individuals. Constant individuals might also be excluded, depending on the control problem. Non-trivial individuals may be excluded if typical sensor readings lead to function values or function behaviors which are clearly out of bound in terms of amplitude or frequency content. The delicate aspect is that the sensor readings change under actuation and a good guess of the probability distribution of sensor readings $p(\mathbf{s})$ is required.

Another strategy to avoid the testing of non-promising individuals is to reduce the number of operations per individuals for instance by low initial tree depth for the first generation.

Advanced material 7.1 Pre-evaluation in `OpenMLC`.

A more proactive measure can be undertaken by the use of a pre-evaluation function. This function takes the individual as an argument and returns a simple logical. If 'false' is returned, the individual is discarded and a new one is generated (or evolved). Detection of zero-individuals can usually be easily achieved with a simple simplification of the LISP expression (`OpenMLC` provides the function `simplify_my_LISP.m`). Saturated control laws can be detected by providing random time series' covering the whole sensor range for all sensors and discarding the control laws that stay constant (or keep the same value more than a pre-determined percentage of the total samples).

7.5 The imperfect experiment

Finally, we discuss the performance of MLC under imperfections of the experiments, like high-frequency noise (Sec. 7.5.1), low-frequency drift (Sec. 7.5.2), and the need to monitor the experiment's health status (Sec. 7.5.3).

7.5.1 Noise

In turbulence, noise is not an imperfection, but an intrinsic feature. For the purpose of control, we consider noise as the high-frequency fluctuations which are not relevant for the actuation mechanism. Noise impacts two aspects of MLC, the actuation command and the cost function. The actuation command $\mathbf{b} = \mathbf{K}(\mathbf{s})$ is affected by noise in the sensor signals. Every operation in the expression tree for $\mathbf{K}$ can act as noise amplifier and at some complexity the control law may become a noise generator. Hence, the number of operations or the depth of expression tree needs to be limited.

Second, the impact of noise on the cost function is far less critical, as the corresponding integrand is generally simple and the noise is largely integrated out.

The use of filters may improve MLC performance. Examples are low-pass, band-pass or single-frequency Morlet filters. We emphasize, again, that any predetermined filter introduces a bias towards the expected actuation mechanism which is not necessarily the most effective one. The choice of the proper filter may also be integrated in the MLC search algorithm like in Chapter 4.

7.5.2 Drift

Low-frequency drifts may arise due to changing operating conditions, e.g. changing oncoming velocity, temperature change or varying hot-wire anemometer readings. Drifts imply that effective individuals may degrade in performance and ineffective individuals may improve. Drifts may also lead to the choice of different sensors in the best control laws. However, within one or few generations, MLC will adjust to the new operating conditions if the change in performance of a given individual is slow with respect to the testing time of one generation. If the change is significantly faster, the cost function values of a single generation are wrong and lead inevitably to a poor ordering.

The cure for drifts depends on the cause. If the sensor readings drift, a recalibration in regular intervals is advised. If the cost function drifts, the cure may be a normalization with the flow response to a periodic forcing. If the drifts are measured, like the oncoming velocity, the parameter may be integrated into the control law. We shall not pause to give a more exhausting list of drifts and potential cures.

7.5.3 Monitoring

Finally, a MLC run may be a time-consuming experiment with complex installations. Apart from slow drifts, there may be sudden unintended changes of the equipment. One or several hot-wire sensors may break. A pressure sensor may get clogged. An actuator may fail. Or there may be a sudden drop in the pressure reservoir of the actuator leading to significantly reduced actuator efficiency. Parts of equipment may unintentionally vibrate for some control laws. And the list can be indefinitely continued. Hence, an automatic extensive health monitoring of the plant is strongly advised to avoid extensive time on unsatisfying results. The authors have experienced several times that MLC has exploited weaknesses of problem definition, the wind tunnel or the actuator setup. It is a common experience that control optimization exploits the weakest links of the theoretical problem formulation / contraints or imperfections of the plant.