Kernel Filter-Based Adaptive Controllers For Cybergenetics Applications

Benjamin Smart[1], Lucia Marucci[2,3,4,*], Ludovic Renson[1,*¶]

October 18, 2024

Abstract

Cybergenetics is an advancing field seeking to implement control theory within biological systems. When applying feedback control for the regulation of gene expression or cell proliferation, model-based control strategies can be applied; in this context, online adaptive mathematical models can be used to keep models in tune with the current behaviour of the biological system. Controllers are often constrained by their sampling rate, which is usually relatively low when using microfluidics/microscopy platforms. Current adaptive filters can lead to an inaccurate predictive model when operating with a low sampling rate, leading to sub-optimal control. Here, we propose a kernel filter that can adapt model parameters online to produce a more accurate predictive model that can be included within an adaptive model predictive control scheme. The use of the kernel filter is demonstrated in *in silico* experiments, where we control a synthetic gene network oscillator and a P53 oscillator. Results show that the kernel filter outperforms a particle filter when used for parameter estimation in both the predictive model accuracy and when included within an adaptive model-based controller.

[*1]Department of Mechanical Engineering, Imperial College London, London SW7 1AY, U.K. b.smart21@imperial.ac.uk l.renson@imperial.ac.uk

[†2]School of Engineering Mathematics and Technology, University of Bristol, Ada Lovelace Building, University Walk, Bristol BS81TW, U.K.

[‡3]BrisSynBio, Life Sciences Building, University of Bristol, Tyndall Avenue, Bristol BS8 1TQ, U.K.

[§4]School of Cellular and Molecular Medicine, University of Bristol, University Walk, Bristol BS8 1TD, U.K. lucia.marucci@bristol.ac.uk

[¶*] L.M. and L.R. are co-last and co-corresponding authors.

1 Introduction

Cybergenetics is a recent field coming out of synthetic biology, which applies principles and techniques from control engineering to forward engineer complex phenotypes in living cells, such as gene expression and cell proliferation [1, 2]. Model-based controllers can be used in cybergenetics [3, 4], often offering better performance than model-free strategies.

Parameter estimation is vital to all system engineering problems requiring a mathematical model [5]. Most biological control applications predominantly rely on models with fixed parameters, i.e. parameters estimated offline using calibration data sets [6, 7]. However, when performing closed-loop experiments, the system of interest may be pushed into a new area of the problem space where the identified parameters can fail to describe its behaviour [8]. It is indeed well known that mathematical models of biological systems can suffer from identifiability issues [9] because of the impossibility of directly measuring most of the parameters and of noisy behaviours, in addition to possible uncertainties in model structure [10]. Therefore, it is beneficial that online parameter estimation is included within cellular model-based feedback control schemes.

Online parameter estimation combines the benefits of physics-based and data-driven control, utilising some of the stability guarantees and robustness of the model-based controllers whilst avoiding sensitivity problems due to the uncertainties in fixed model parameters and any noise/variance in observations, [11, 12].

Regarding the experimental implementation, cybergenetics mainly relies on the use of microscopy/microfluidics platforms, where time-lapse imaging is combined with image segmentation [13] to quantify real-time outputs (for example gene expression), compute the control error by comparing the output to the target behaviour, and finally apply control algorithms which can provide, to cells, control inputs to minimise the control error [14, 15, 16, 17, 18, 19]. The control algorithms can include the online adaption of models, often using particle filters [20]. The sampling rate at which images are collected and analysed is usually limited because of the physical constraints of the microscope and, most importantly, issues related to phototoxicity when performing long experiments. Therefore, there is a need for adaptive models that can be derived at a lower sampling rate. Particle filters are frequently used to extract biological system state and parameter information offline, [21, 22] and online [7, 23]. Recently, particle filters are also being used within feedback controllers working on biological systems[24, 20].

Here, we propose a kernel filter that can adapt model parameters at low sampling rates and be included within an adaptive model predictive controller (MPC). The kernel filter proposed here is adapted from [25]. [25] demonstrates parameter estimation offline on a mechanical system. In this paper, we extend this to use the kernel filter online, using a shifting kernel window and a shifting sampling grid to solve the nonlinear optimisation problem online, updating the system parameters and kernel hyperparameters each sampling period.

Many biological systems present nonlinear oscillatory dynamics, occurring across temporal and spatial scales down to cellular cycles with millisecond time periods [26]. The first example considered here is a synthetic gene network oscillator implemented in bacterial cells ([27]. The second example is the P53 biological oscillator [28], found in humans, which controls several critical cellular processes, including cellular senescence, apoptosis, cell cycle arrest and DNA repair. Of note is that the two systems present very different dynamics (the bacterial oscillator is an order of magnitude faster than the mammalian one). We observe that the kernel filter outperforms a particle filter when a low sampling rate (less than five samples per period) is used. We also show that, for the synthetic gene network oscillator, the kernel filter is better over all sampling rates.

The structure of this paper is as follows. Section 2 describes the kernel and particle filters, including an overview of the filters used within a feedback controller using MPC. Section 3 compares the performance of the two filters in predicting the future behaviour of the states and compares the performance within an MPC controller for the two oscillators. Section 4 presents the conclusions of this study.

2 Methods

2.1 Kernel Filter

Consider the dynamic system

$$\dot{\mathbf{x}} = f(\phi, \mathbf{x}), \qquad y = \mathbf{x} + d(\xi), \tag{1}$$

where $\mathbf{x}$ are the states of the system, f is a known nonlinear vector field and ϕ are the unknown model parameters. y represents the measurement of the states, which is assumed to be perturbed by additive Gaussian noise, d, of covariance, ξ. The state estimate at time t, $\hat{\mathbf{x}}(t)$, is given by solving

$$\dot{\hat{\mathbf{x}}} = f(\hat{\phi}, \hat{\mathbf{x}}), \tag{2}$$

where $\hat{\phi}$ are the estimated model parameters.

kernels are frequently used to predict a future state [29], by lifting lower dimensional nonlinear problem space measurements, y, to infinite dimensional linear feature spaces. The kernel is then used to describe the cross product of two measurements in this Hilbert space, meaning that the infinite transformation dictionary never has to be evaluated. Rather than comparing a single measurement to a model prediction, the kernel can be used to relate all previous measurements and indicate whether the prediction from a candidate model seems reasonable when compared to all the past measurements. In Ref. [25], the authors estimate the nominal parameter set, ϕ, by maximising the similarity between a kernel containing the outputs of a candidate model and a kernel containing just the

measurements, using the Hilbert Schmidt Independence Criterion (HSIC) [30]. Here, we adapt this method to work online, to be part of an adaptive MPC re-estimating the system parameters within each time step and using a sliding kernel window and an adaptive grid search.

We define the measurement vector, $Y_{t-W:t} = [y^T(t-W), y^T(t-W+1), ..., y^T(t)]$ and a state vector, $X_{t-W:t} = [\mathbf{x}^T(t-W), \mathbf{x}^T(t-W+1), ..., \mathbf{x}^T(t)]$, where W represents the kernel's sliding window size. W prevents the kernel from growing in size as more measurements are taken.

The prediction kernel, $\kappa^x(\hat{\phi})$, uses a mixed vector, $Z_t(\hat{X}_{t,1}(\hat{\phi}), Y_{t-W:t-1}) = \begin{bmatrix} Y_{t-W:t-1} & \hat{X}_{t,1}(\hat{\phi}) \end{bmatrix}$ containing past measurements and one predictive step, $\hat{X}_{t,\upsilon}(\hat{\phi})$, where υ is the step number within the prediction horizon used. Following investigations, a Periodic kernel is used in all simulations as discussed in Section S3. Each element of the kernel is calculated as

$$\kappa^x_{i,j}(Z_t) = exp\left(-\frac{4}{\ell} sin\left(\frac{\pi(Z_i - Z_j)}{p}\right)\right) + \sigma_n. \tag{3}$$

The measurement kernel, κ^y, is calculated using only the measurements $Y_{t-W:t}$ as

$$\kappa^y_{i,j}(Y_{t-W:t}) = exp\left(-\frac{4}{\ell} sin\left(\frac{\pi(Y_i - Y_j)}{p}\right)\right) + \sigma_n, \tag{4}$$

where p is the periodic hyperparameter, ℓ dictates the exponential length scale of the kernel and σ_n weighs the expected noise. The hyperparameters are initially chosen by the method described in [25], and then optimised, online, using Marginal Likelihood, [31], throughout the first kernel window. Hyperparameter optimisation is further discussed in Section S4.

The similarity between the predicted kernel and measured kernel spaces is compared using the Hilbert Schmidt Independence Criterion (HSIC) [30],

$$HSIC(\kappa^x, \kappa^y) = \left[\frac{\mathrm{Tr}(\kappa^x H \kappa^y H)}{||\kappa^x H||_F^2 ||\kappa^y H||_F^2}\right], \tag{5}$$

where $H_{i,j} = \xi_{i,j} - \frac{\Delta t}{W}$, ξ is the covariance of the noise present (assumed to be white noise, ξ is a diagonal matrix) and Δt is the time difference between samples. The score tends to one if the two kernels are identical and tends to zero as the difference increases. The global maximum of the HSIC would correspond to the nominal parameter set, ϕ.

To find $\hat{\phi} = \phi$, Ref. [25] implements both a grid search and a nonlinear optimiser algorithm. Their grid search is a static offline random sample search. Here, we implement a moving sample window around the operating point, as outlined in Algorithm 1, so the filter can operate online. Gradient-decent-based nonlinear solvers (MATLAB's® fmincon, IPA, and NOMAD [32]) were not used as it was found that they often settle on local minima near their initiation point, not improving the predictive states.

A grid search requires some knowledge of the initial parameter values to define the finite grid and is otherwise very computationally heavy. The grid used here trials three different values for each parameter, one at the previous optimal parameter value, $\hat{\phi}_{t-1}$, and another two at $\pm\rho\%$. The possible parameter change between any two samples depends on the sampling rate. Therefore, the search grid intervals, ρ, should be dependent on this sampling rate. This relationship is discussed in Section S5. For a system containing n parameters, the discrete set, Φ_t, contains all 3^n possible combinations of $\tilde{\phi}_t$ where each element of $\hat{\phi}_{t-1}$ has been multiplied by an element of $([1-\rho, 1, 1+\rho])$ to create a 3^n unique trial $\tilde{\phi}_t$. Whichever sample set, $\tilde{\phi}_t$, produces the highest HSIC will be chosen as the kernel filter's parameter estimate, $\hat{\phi}_t$. The grid size is defined relative to the parameter values to follow the observed parameter changes. A finer grid can be used, resulting in higher computational costs.

Algorithm 1 Kernel Filter

for each sample, at time, t **do**
  Create the Measurement kernel for the window, W
  if $t < W$ **then**
    Optimise kernel hyperparamters, p, ℓ and σ_n
    Evaluate $\kappa^y(Y_{0:t})$ in Eq.(4)
  else
    Evaluate $\kappa^y(Y_{t-W:t})$ in Eq.(4)
  end if
  Try all 3^n trial parameter sets, $\hat{\phi}$
  for each $\tilde{\phi}_t \in \Phi_t$, **do**
    Predict $\tilde{X}_t = f(\tilde{\phi}_t, Y_{t-1})$ in Eq.(2)
    Create the Prediction kernel for the window, W
    if $t < W$ **then**
      Evaluate $\kappa^x(Z_t(\tilde{X}_t, Y_{0:t-1}))$ in Eq.(3)
    else
      Evaluate $\kappa^x(Z_t(\tilde{X}_t, Y_{t-W:t-1}))$ in Eq.(3)
    end if
    Evaluate and Store $HSIC(\kappa^x, \kappa^y)$ in Eq.(5)
  end for
  $\hat{\phi}_t = \arg\max_{\tilde{\phi}_t} HSIC(\kappa^x, \kappa^y)$
end for

2.2 Particle Filter

Particle Filters are a class of Bayesian filters that estimate the parameter distributions based on output observations [33]. It is a well-established method for adaptive models and is included here for comparison. The particle filter aims to find $P(\phi|Y_{0:t})$, the probability of the parameter set given the past measurements. The first two are the basis of all Bayesian filters, using Bayes theorem to state an a priori about the distribution of the parameters, $P(\phi)$, and the observations of the output to measure, $P(Y_{0:t}|\phi)$. Marginalisation alongside marginal likelihood is used to approximate the probability of the output, regardless of the parameter set, $P(Y_{0:t})$, by propagating

M sample parameter sets, $\tilde{\phi}$, through the system, $\tilde{X}_t = f(\tilde{\phi}, Y_{0:t-1})$. The distribution of the output, $P(Y_{0:t}|\phi)$, can be approximated as the distribution of these samples, $P(\tilde{X}_{0:t}|\tilde{\phi})$. Each parameter set, $\tilde{\phi}_i$, is weighted by ω_i, an estimate of how likely its output, $\tilde{X}_{0:t}$ would be to occur for the given output distribution, $P(Y_{0:t}|\phi)$. Each ω is summed to form an a posteriori parameter distribution,

$$P(\phi|Y_{0:t}) \approx \sum_{i=1}^{M} [\omega_i \delta(\phi - \tilde{\phi}_i)], \tag{6}$$

where δ is the Dirac function. With the estimated parameter set as the expected value of this distribution, $\hat{\phi} = E(\phi|Y_{0:t})$.

The final step of a particle filter utilises an importance distribution, scaling the weighting, ω, by how accurate our model is perceived to be, knowing that it is not an exact representation of the system. After each iteration, the particles can be resampled. The particle filter simulation parameters are discussed in Section S6.

2.3 State Error Index

To assess the error of the predictive model adapted by the filters, we define the identification Mean Absolute Error, MAE [34], as

$$MAE = \frac{1}{T\bar{N}} \left[\sum_{i=1}^{T} \sum_{j=1}^{\bar{N}} ||\hat{X}_{i,j} - X_{i+j}||^1 \right], \tag{7}$$

calculating the sum over the total simulated time, T, of the normalised absolute difference between the predictive states, $\hat{X}$, and the actual states X, over an N step prediction horizon. $\bar{N} = \frac{N}{\Delta t}$, the total number of samples within the N-step horizon.

2.4 Feedback Control Method - Adaptive Model Predictive Control (MPC)

MPC uses a model of the plant (system to be controlled) in a feedback loop to estimate the effect of future control inputs on the states, N time steps into the future. Prediction horizon steps, N, are discrete time steps in the unit of the model (minutes for the bacterial system in Section 3.1 and hours for the mammalian system in Section 3.2), evaluating the predicted state at each step. These control inputs are chosen to minimise a cost function, $J(\mathbf{E}, \mathbf{U})$, along with any control input or state constraints [35],

$$\begin{aligned} &\min_{\mathbf{U}} (J(\mathbf{E}, \mathbf{U}) = \mathbf{E}^T \mathbf{Q} \mathbf{E} + \mathbf{U}^T \mathbf{R} \mathbf{U}) \\ &st. \quad -\mathbf{U} \leq \mathbf{U}_L, \quad \mathbf{U} \leq \mathbf{U}_U, \end{aligned} \tag{8}$$

MATLAB's® nonlinear solver, 'fmincon', solves the nonlinear optimisation problem with its interior point algorithm, limited to ten iterations. $\mathbf{E} = [\mathbf{e}(0), \mathbf{e}(1), \ldots, \mathbf{e}(N)]^T$ describes the future predicted state errors using control inputs, $\mathbf{U} = [\mathbf{u}(1), \mathbf{u}(2), \ldots, \mathbf{u}(N)]^T$ in the estimate model of the system. The updated model parameters are provided by either the kernel filter or the particle filter in each iteration (outlined in Section 1). Therefore, the model within the MPC adapts online to the observations being made.

The weight of each cost function term can vary the optimal control input. In the examples described here, the control inputs have to stay within their bounds, and only the tracking error is costed ($\mathbf{R} = 0$, and $\mathbf{Q} = 1$). In all examples, up to the first ten future control inputs are added to the system to reduce the computation cost and increase the effect of the model on control performance.

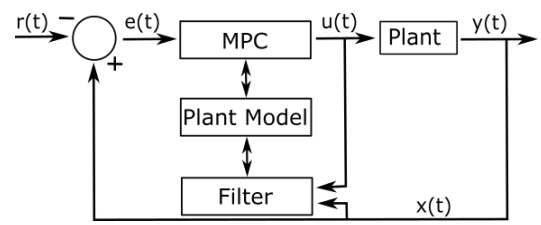

Figure 1: The MPC architecture. The biological system to be controlled, the plant, is observed in the output, y. The model states, x, are assumed to coincide with the plant outputs y. x is given to the filter to adapt the plant model and compared to the reference, r, to be given to the MPC algorithm. The MPC optimally chooses the input, u, which acts on the plant system.

2.5 Control Error Index

The performance of the controller is taken as the error between the output state and its reference; we define the control Root Mean Squared Error, $RMSE$, as

$$RMSE = \frac{1}{T} \sqrt{\sum_{i=1}^{T} \left(Y_i - r_i\right)^2}, \tag{9}$$

calculating the sum over all time, T, of the difference between the actual output Y and reference, $r_i = r(t = i\Delta t)$.

3 Results

3.1 Synthetic Gene Network

The model of synthetic gene network found in [27] has been used here, describing the interactions between a CI protein (X_1) and a LAC protein (X_2). A four-state kinetics model, containing both the proteins and their corresponding dimers, is reduced to a two-state model containing a dimensionless representation of the CI protein (x_1) and the LAC protein as (x_2),

$$\dot{x}_1 = \frac{1+x_1^2+\alpha\sigma x_1^4}{(1+x_1^2+\sigma x_1^4)(1+x_2^4)} - u_{x_1}x_1, \tag{10}$$

$$\tau\dot{x}_2 = \frac{1+x_1^2+\alpha\sigma x_1^4}{(1+x_1^2+\sigma x_1^4)(1+x_2^4)} - u_{x_2}x_2 \tag{11}$$

$$y = x_1 + d(\xi). \tag{12}$$

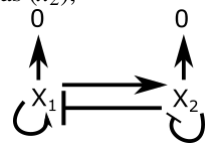

Figure 2: The synthetic gene network in [27]. The schematic shows the interactions between a CI protein (x_1) and a LAC protein (x_2); arrow-ended lines indicate activation, and bar-ended lines indicate inhibition.

The CI protein promotes both the CI and LAC dimer, whereas the LAC protein inhibits both dimers, as shown in Fig. 2. The system parameters are discussed in Section S1. For all time varying parameter simulations, $\phi_0 = [\alpha, \sigma] = [11, 2]$.

3.1.1 Low sampling rate filtering

To compare the error of the kernel filter to the commonly used particle filter, [36], α and σ within the nominal parameter set, ϕ, has been changed throughout a simulation. This change can be seen by the black lines in Fig. 3B, with an increase in α to 2.25 times its initial value, then an equivalent decrease, followed by an increase of σ to 2.25 times its initial value, then an equivalent decrease. Within the nominal parameter set, ϕ, all other parameters remain at their initial values throughout the simulation. The filters attempt to adapt the parameter set, $\hat{\phi}$, such that state predictions, $\hat{X}(\hat{\phi})$, follow the actual states, $X(\phi)$. The prediction trajectories show the state trajectories that are predicted using the state and estimated parameters. Predictions are made at each output sample.

Fig. 3A displays the predicted trajectories when the system is sampled every 12.5 minutes. The kernel filter manages to change the amplitude and period of the oscillations between 300 and 900 minutes to almost no oscillations. In contrast, the particle filter struggles to remain close to the response of the system simulated with nominal parameters' values. The identification *MAE* reflects this as $0.0884 < 0.2850$ for the kernel and particle filter, respectively.

Fig. 3B shows the parameter variation. It can be seen that the kernel filter makes up for the decrease in α by initially increasing σ, still achieving a reasonable state estimation. It is also shown that the particle filter does not significantly adapt either parameter from the initial estimation. The model is over-parameterised, resulting in parameter non-identifiability [37], meaning that there are several parameter sets, $\bar{\phi}$, that lead to near identical state predictions and therefore, the filter could choose any of these sets as $\hat{\phi}$ and still achieve good state tracking. Section S7 includes simulations with measurement noise of covariance $\xi = 10^{-2.5}$.

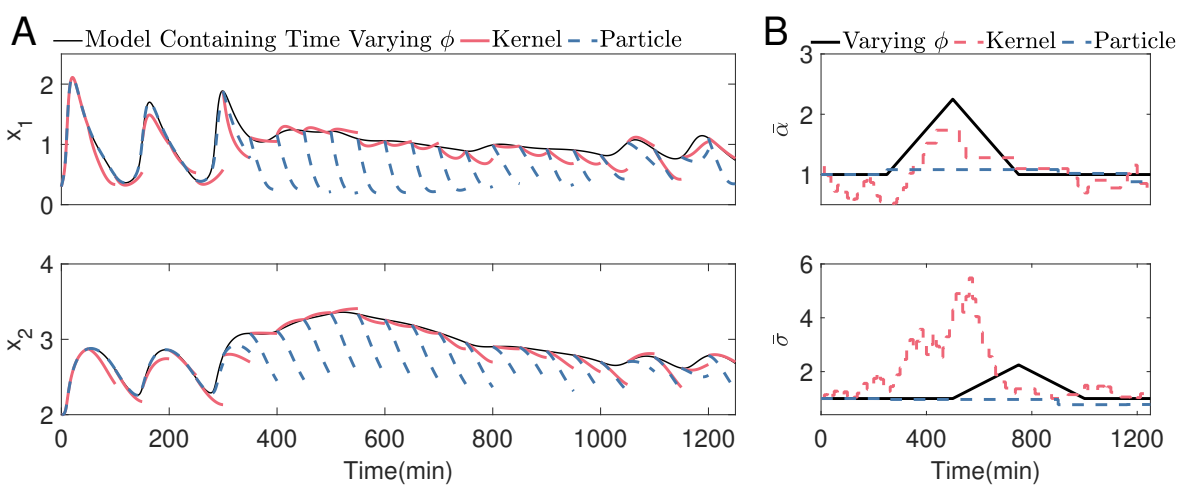

Figure 3: The filters adapting the parameters, online, to match states of the synthetic gene network (Eq.(10) - Eq.(11)) in which the nominal parameter value, ϕ, changes over time. One sample every 12.5 minutes (8.8 samples per period) is used, which is considered a low sampling regime here. Other simulation parameters: $\frac{W}{\Delta t} = 500$, $M = 5000$, 0.5 re-sampling threshold, 0.0158 expected variance of parameters. A) Every 50 minutes, the prediction trajectory, $\hat{X}_{t,1:50}$ of each filter is compared to the nominal model. B) The parameter adaptation of $\hat{\phi}$, compared to the nominal parameters, ϕ. Each parameter has been normalised by the initial parameter defined in Section 3.1.

3.1.2 Low sampling rate control

As discussed in Section 2.4, either filter can be used to adapt the model within an MPC scheme. The synthetic gene network has been synthetically designed to oscillate in its free response [38]; the controller discussed here can be used to steer the amplitude and period of the oscillations. A sine wave with a frequency of roughly ten times slower than the free response is used as the reference, $\mathbf{r}(t)$. These slow reference dynamics can highlight any errors within the MPC's model.

Identical to Section 3.1.1, the temporal evolution of α and σ within the nominal parameter set, ϕ, has been changed throughout a simulation, shown by the black lines in Fig. 4B. The filters attempt to adapt the parameters, $\hat{\phi}$, such that state predictions, $\hat{X}(\hat{\phi})$, follow the actual states, $X(\phi)$ and therefore give the MPC algorithm a more accurate model.

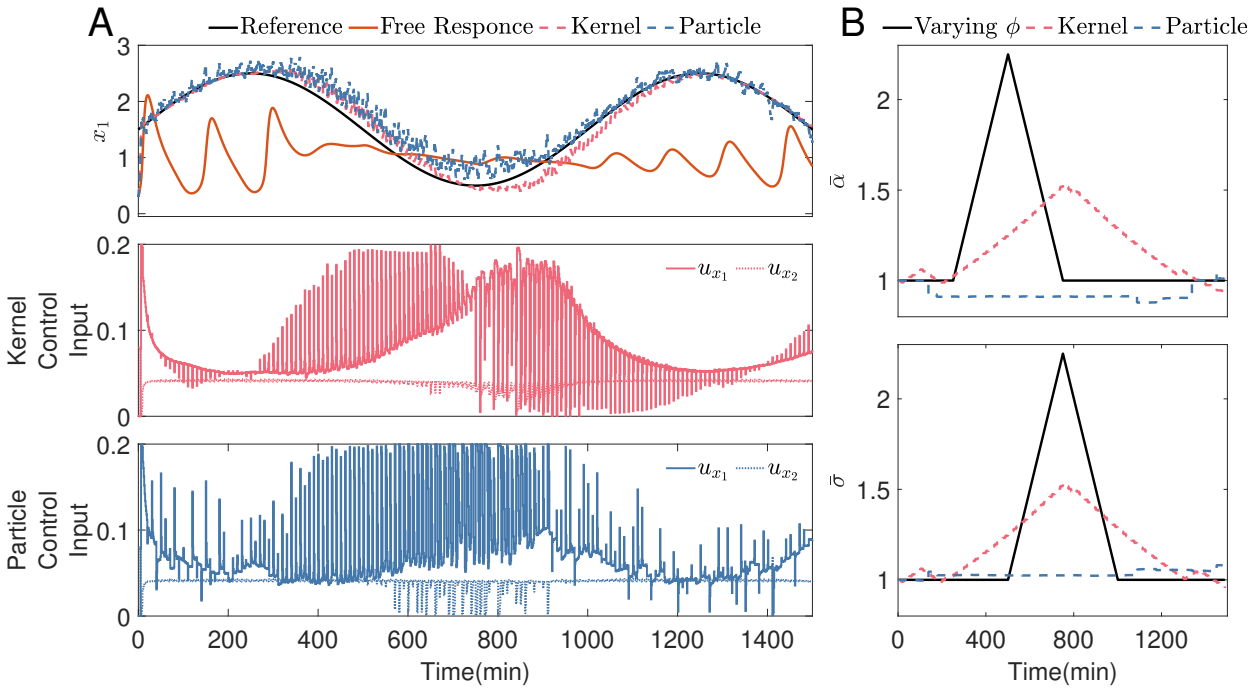

Figure 4: The filters adapting the parameters, online, to match the states of the synthetic gene system (Eq.(10) - Eq.(11)) in which the nominal parameter value, ϕ, changes over time. One sample every 12.5 minutes (8.8 samples per period) is used, which is considered a low sampling regime here. Other simulation parameters: the control input is actuated once a minute; $N = 10$ minutes, containing ten input steps $\frac{W}{\Delta t} = 50$, 0.0001 is the expected variance of parameters, $M = 5000$, 0.5 resampling threshold. A) The system's output qualitatively follows the sine wave reference using the control input shown. B) The parameter adaptation of $\hat{\phi}$, compared to the nominal parameters, ϕ. Each parameter has been normalised by the initial parameter defined in Section 3.1.

Fig. 4A shows the synthetic gene network control response sampling every 12.5 minutes (8.8 samples per period). It can be seen that when the free response amplitude and centre of oscillation change between 400 and 1000 minutes, the kernel filter has a smaller control error than the particle filter. The particle filter's MPC believes the system would naturally oscillate with larger amplitudes between these times, causing the controller to miss the reference. The control *RMSE* of the synthetic gene network is 0.1529 for the kernel filter and 0.2330 for the particle filter.

Fig. 4B shows the parameter variation of the synthetic gene network sampling every 12.5 minutes. The particle filter rarely changes the parameters, whereas the kernel filter increases α and σ when α increases in the nominal set. These parameter changes lead to the smaller control error seen in Fig. 4A.

3.1.3 Varied sampling rate identification and control

The identification Mean Absolute Error *MAE* Eq.(7) of the two filters is compared over a range of sampling rates in Fig. 5A, looking at the prediction trajectories (as in Fig. 3). The control Root Mean Squared Error, *RMSE* Eq.(9) of the two filters is compared over a range of sampling rates in Fig. 5B comparing the control performance of the two filters following the same reference as described in Section 3.1.2, used within MPC controllers (as in Fig. 4). It can be seen that the kernel filter outperforms the standard particle filter over all sampling rates up until 80 minutes (1.57 samples per period), where both filters follow the identification *MAE* of the static model.

Similarly to the identification MAE in Fig. 5A, the control *RMSE* in Fig. 5B shows the kernel filter outperforming the particle filter over all sampling rates.

For fast sampling rates, it is expected that the identification *MAE* and control *RMSE* would be reduced as the filters are observing the system more frequently. This can be seen for both the particle and the kernel filter. Once the filter is used in a rapid sampling regime, the best filter will depend on the filter setup and the underlying system itself. Depending on the relative size of the parameters and the output's sensitivity to each parameter.

The control *RMSE* can be seen in Fig. 5C and D over varied sampling rates and prediction horizons, N. For both the particle and kernel filters, it can be seen that as the prediction horizon gets longer, the control error increases. We expect that there will be a decrease in tracking error with longer prediction horizons as the controller gains more information about the future consequences of the inputs. However, for nonlinear systems with time-varying parameters, longer predictions with fixed parameters, $\hat{\phi}$, can mislead the MPC [39]. Model prediction trajectories consider fixed parameters, $\hat{\phi}$, which are only locally accurate. A significant decrease in control *RMSE* can be seen when including a second time step in the prediction horizon for slower sampling times. This is explained by the fact that each prediction step represents a significant jump in time for each state at such a low sampling rate. For instance, for 80 minutes between samples, one prediction step represents almost 0.75 times the oscillation period of the system.

As the sampling rate decreases, the control error increases. Therefore, we expect the bottom left corner of the plot to be the darkest region, with the lowest control error and the top right corner to be the brightest, with the highest control error. For the kernel filter in Fig. 5C, for all prediction horizons, the offset of the darker region persists until larger sampling rates are compared to the particle filter in Fig. 5D. There is a significantly darker region for the kernel filter compared to the particle filter between sampling times of 2.5-60 minutes ($44 - 1.83$ samples per period) and any prediction horizon, showing that the kernel filter outperforms the particle filter, not only at $N = 10$ (as in Fig. 5B) but for all prediction horizons shown.

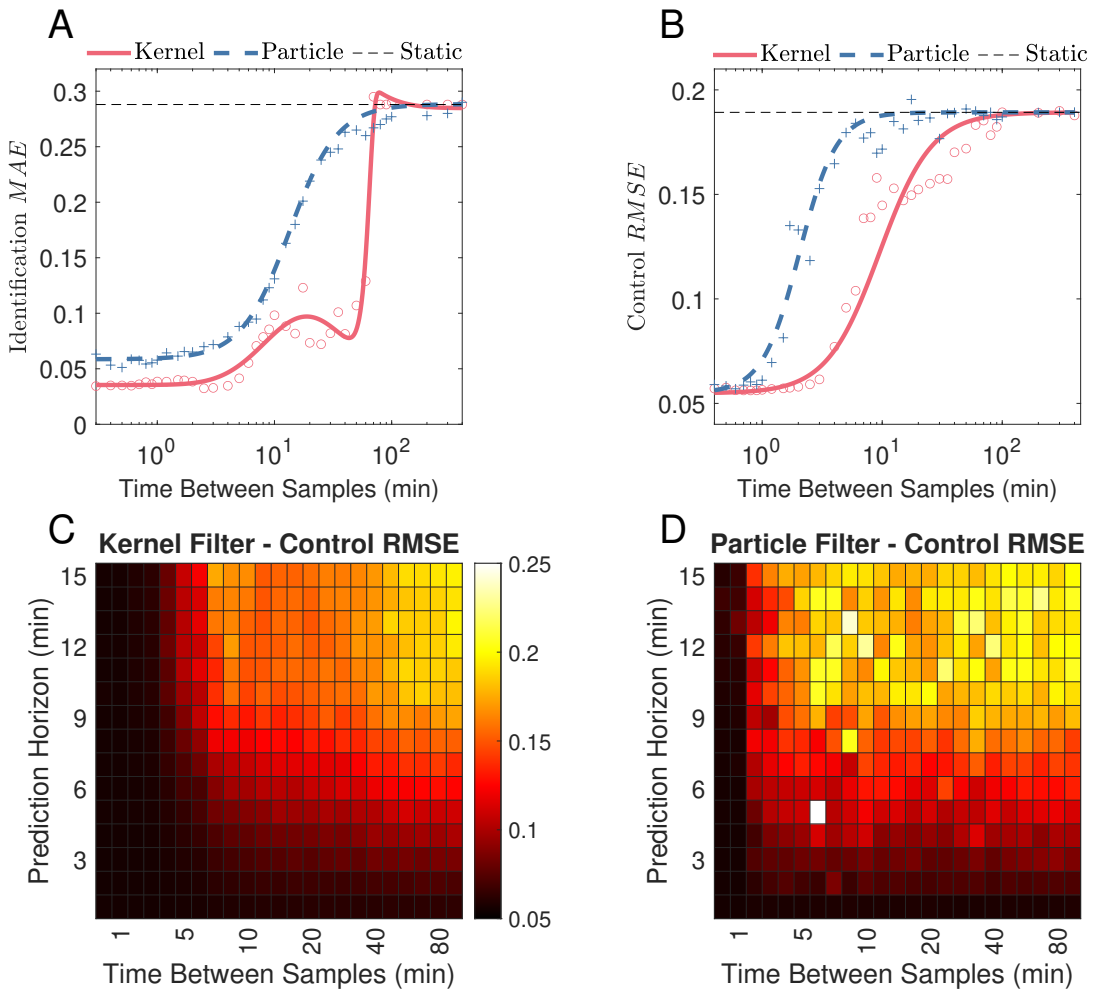

Figure 5: A) The identification Mean Absolute Error (*MAE*, Eq.(7)) of the two filters is compared over a range of sampling rates for the synthetic gene network. Fig. 3 notes the simulation parameters for the kernel and particle filters. B) The control Root Mean Squared Error (*RMSE*, Eq.(9)) of the two filters are compared over a range of sampling rates for the synthetic gene network, with simulation parameters as in Fig. 4. C) and D) The control *RMSE* of the kernel filter(C) and the particle filter(D), compared over a range of sampling rates and prediction horizons, N, for the synthetic gene network. Other simulation parameters are in Fig. 4.

3.2 P53 Oscillator

The model of a P53 oscillator describes the interactions between the p53 protein and the Mdm2 protein. Model 4 from [28] is used here, containing three states: P53 protein as x_1; the Mdm2 precursor as x_2 and the Mdm2 protein as x_3.

$$\dot{x}_1 = \beta_x - \alpha_k x_3 \frac{x_1}{x_1 + k}, \tag{13}$$

$$\dot{x}_2 = \beta_y x_1 - \alpha_0 x_2, \tag{14}$$

$$\dot{x}_3 = \alpha_0 x_2 - \alpha_y x_3 - k_m u x_3, \tag{15}$$

$$y = x_1 + d(\xi). \tag{16}$$

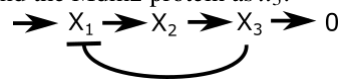

Figure 6: P53 Oscillator formed by negative feedback from an Mdm2 protein (x_3) on a P53 protein (x_1), including a transcriptional Mdm2 precursor (x_2). Arrow-ended lines indicate activation, and bar-ended lines indicate inhibition.

A schematic of the P53-Mdm2 interaction is shown in Fig. 6, displaying this cascade system's feedback through an inhibition of P53 from Mdm2. The input to the system, u, is an extra degradation of Mdm2 [40].

The system parameters are discussed in Section S2. For all time varying parameter simulations, $\phi_0 = [\alpha_y, \alpha_0, \alpha_k, k, \beta_x, \beta_y] = [0.8, 0.9, 1.7, 0.001, 0.9, 1.2]$.

The system has six kinematic constants, and three have been selected to be adapted within the filters. α_0, α_k and β_y have been chosen as they characterise the speed of the two cascade reactions, $x_1 \rightarrow x_2$ and $x_2 \rightarrow x_3$ as well as the feedback inhibition, $x_3 \dashv x_1$.

3.2.1 Low sampling rate filtering

We compare the state trajectory error of the filters as discussed in Section 3.1.1. This change in α_0 and α_k can be seen by the black lines in Fig. 7B, with the increase in α_0 to 1.5 times its initial value then an equivalent decrease, followed by a decrease of α_k to 0.5 times its initial value, then an equivalent increase. The prediction trajectories are the future predictions using the current model taken from the point where every fourth measurement is taken. The trajectories are 10 hours long.

The predicted states of the P53 system, sampled every 2 hours (3 samples per period) in Fig. 7A, demonstrate that the kernel filter outperforms the particle filter. The trajectory length is 10 hours (every fifth measurement), to increase the filter error and make a clear, visible, distinction between the two filters' performances. This can be seen between 25 and 100 hours with the decrease and then increase in amplitude of x_2 and between 125 and 225 hours where neither filter perfectly matches the changes in the centre of the oscillation for all three states, but the kernel filter does qualitatively and quantitatively better. It can later be seen that this difference is significant in the control simulations between these times. The identification *MAE* (Eq.(7)) of the kernel filter, 0.0685, is significantly less than the identification *MAE* of the particle filter, 0.1760. Section S7 includes simulated measurement noise.

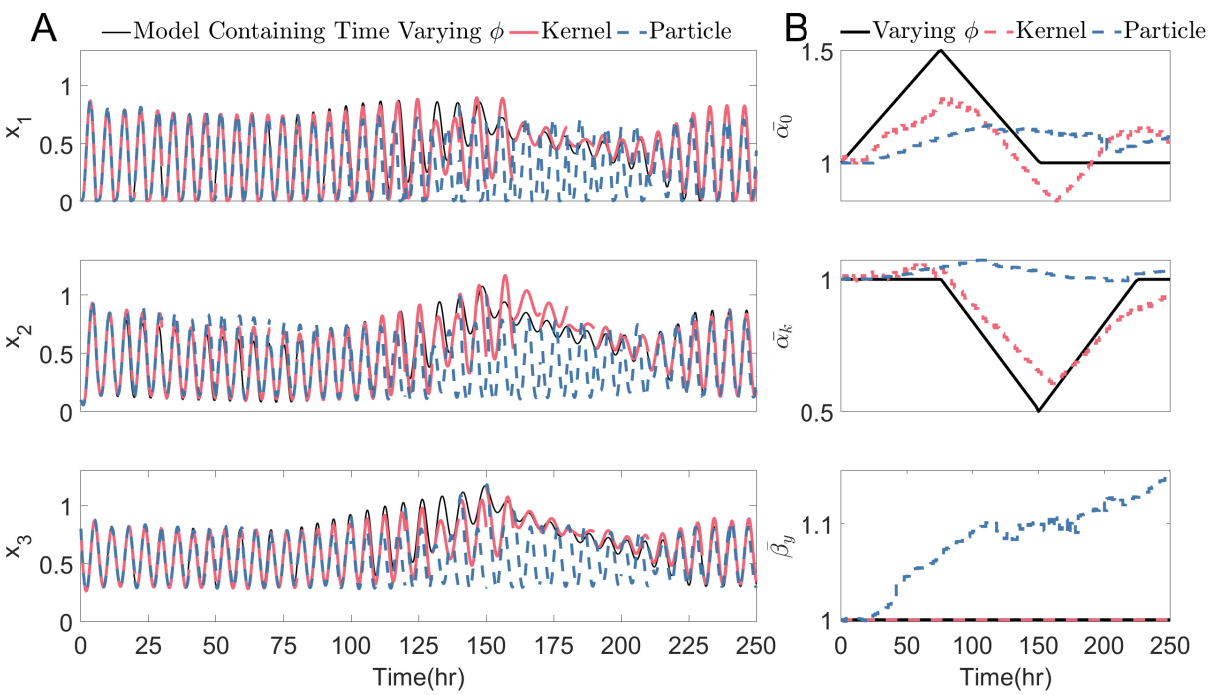

Figure 7: The filters adapting the parameters, online, to match the states of the P53 system (Eq.(13) - Eq.(15)) in which the nominal parameter value, ϕ, changes over time. A sample every 2 hours (3 samples per period) is used, considered a low sampling regime here. Other simulation parameters: $\frac{W}{\Delta t} = 50$, 0.0158 is the expected variance of parameters, $M = 5000$, 0.5 re-sampling threshold. A) Every 10 hours, the prediction trajectory, $\hat{X}_{t,1:10}$ of each filter is compared to the nominal model. B) The parameter adaptation of $\hat{\phi}$, compared to the nominal parameters, ϕ. Each parameter has been normalised by the initial parameter defined in Section 3.2.

The corresponding parameter variations are shown in Fig. 7B. It can be seen that the kernel filter has minimal variation in β_y whilst following the overall trend of α_0 and α_k. The particle filter fails to track these parameter changes, veering all three parameters away from the initial estimate without following the trend of the nominal varying parameters, ϕ.

3.2.2 Low sampling rate control

The P53 system is said to synchronise the cellular clock [41], controlling the proliferation cycle. Even in the presence of genetic mutations, this controller could control the amplitude and period of these oscillations to steer cellular senescence, apoptosis, cell cycle arrest and DNA repair [42], leading to effective personalised cancer treatments and regulation of cellular stress. A sine wave with a frequency of roughly ten times slower than the system's free response is used again as the reference, $\mathbf{r}(t)$. The changes in α_0 and α_k are identical to those discussed in Section 3.2.1.

The control performance of both filters can be seen in Fig. 8A, sampling every 2 hours (3 samples per period). Between 125 and 225 hours, the parameter changes cause the free response to change its centre of oscillation and amplitude. Both filters adapt their models to keep up with this, but it can be seen that the particle filter does not keep up with these changes, causing a large error from the reference between these times. The kernel's control *RMSE* of 0.2297 is less than the particle filter's control *RMSE* of 0.3255.

The parameter changes within Fig. 8B for the P53 system show the kernel filter roughly follows the nominal parameter changes, ϕ. Initiating changes in α_0 and α_k to match the time-varying ϕ. The particle filter does not have an informed variation from the initial estimation at this sampling rate and remains close to the initial estimation throughout the simulation.

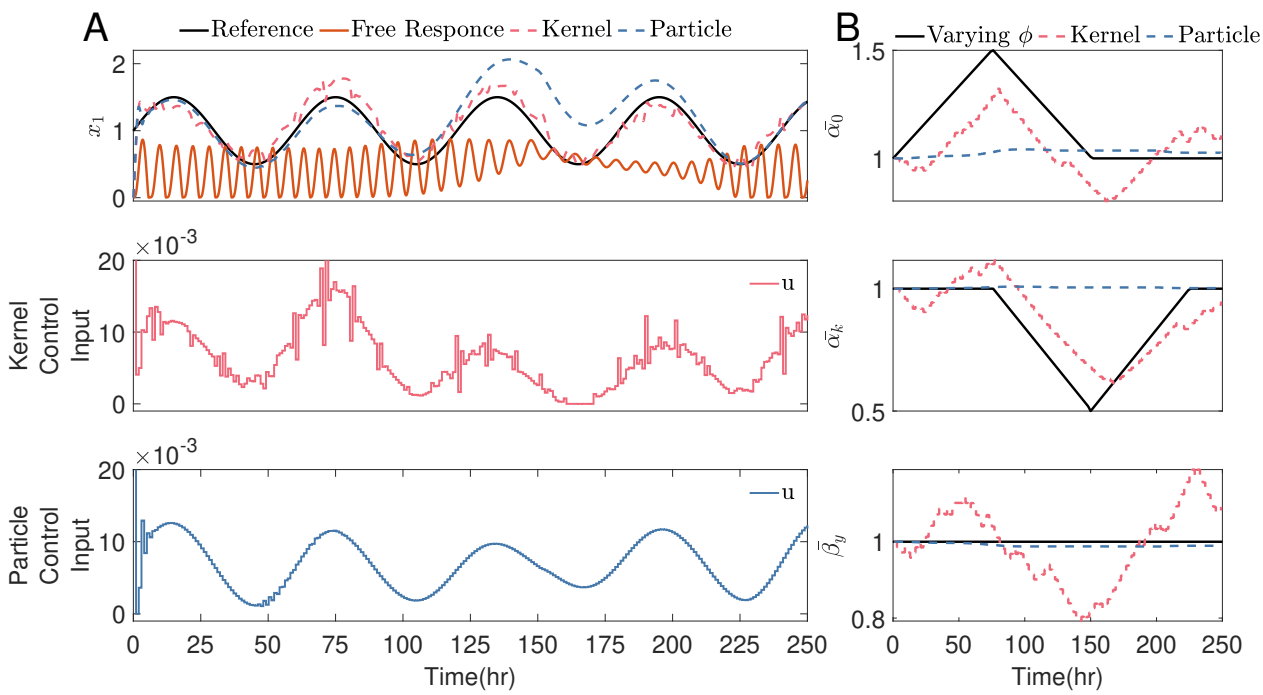

Figure 8: The filters adapting the parameters, online, to match the states of the P53 system (Eq.(13) - Eq.(15)) in which the nominal parameter value, ϕ, changes over time. A sample every 2 hours (3 samples per period) is used, which is considered a low sampling regime here. Other simulation parameters: the control input is actuated once an hour; $N = 10$ hours, containing ten input steps; $\frac{W}{\Delta t} = 50$; 0.0158 is the expected variance of parameters; $M = 5000$, 0.5 resampling threshold. A) The system's output qualitatively follows the sine wave reference using the control input shown. B) The parameter adaptation of $\hat{\phi}$, compared to the nominal parameters, ϕ. Each parameter has been normalised by the initial parameter defined in Section 3.2.

3.2.3 Varied sampling rate identification and control

The identification Mean Absolute Error, *MAE* Eq.(7), indexing the error in the prediction trajectories (as in Fig. 7) and the control Root Mean Squared Error, *RMSE* Eq.(9), indexing the controller performance (as in Fig. 8, using the reference described in Section 3.2.2) are compared over a range of sampling rates in Fig. 9.

For the state trajectories in Fig. 9A, there is a separation of the two filter's identification *MAE* at a sample time of 1.25 hours (5.14 samples per period), where the kernel filter tracking error remains significantly below that of the particle filter, showing that when sampling less than every 1.25 hours, the kernel filter outperforms the particle filter. Sampling less than once every 10 hours (0.6 samples per period), both filters tend to the identification *MAE* of a static model where the parameters are fixed at the initial value. At this point, neither filter varies the parameters, and the initial steady-state oscillation is unaltered by either filter.

There is a transition at 2 hours between samples (3 samples per period), where the control *RMSE* of the kernel filter crosses that of the particle filter for the first time, showing that the kernel filter outperforms the particle filter for sampling times slower than every 2 hours. This transition point (sampling 1.25 times an hour) is similar to the filtering simulation shown in Fig. 9A.

Fig. 9C and D display the control *RMSE* over varied sampling rates and prediction horizons, N. We expect the bottom left corner of the plot to be the darkest region with the lowest control error. There is a darker region for the kernel filter compared to the particle filter between 4 and 7 hour sampling times (1.5-1.17 samples per period) over all prediction horizons, but not between 1 and 3 hours. It can be seen in Fig. 9B that the kernel filter still performs better at slower sampling times not shown in Fig. 9C and D.

The range of sampling times has been chosen to reflect the likely sampling times in-vitro, avoiding phototoxicity of faster sampling times and the slowest sampling times that are well within the device's capabilities. In-silico, we find that the largest performance advantages of the kernel filter are at longer sampling times, up to 40 hours between samples, as discussed in Section S8. This demonstrates the theoretical advantages of the kernel filter within adaptive MPC schemes at even longer sampling times.

Fig. 9C is not as smooth as the synthetic gene network in Fig. 5C; multiple 'light spots' for both the kernel and particle filters can be seen. This difference is due to P53 system dynamics and the increased number of adapted parameters, as discussed in Section S9. The ODE system has six kinematic constants, yet only three have been adapted here. The more parameters added, the lower the average performance error, but the larger the control error variance, as discussed in Section S10. The grid search limits the parameter space that the kernel filter can jump between each step. Therefore, the kernel filter can be trapped in a local minimum for which performance degrades progressively. Without a more global optimisation algorithm, the filter cannot leave this neighbourhood to retrieve more optimal solutions, leading to the 'light spots'. A more extensive grid can reduce a 'light spot' found in Fig. 9C at $N = 9$, sampling time of 2 hours, from $RMSE = 0.3473$ to $RMSE = 0.2365$ (discussed in Section S11).

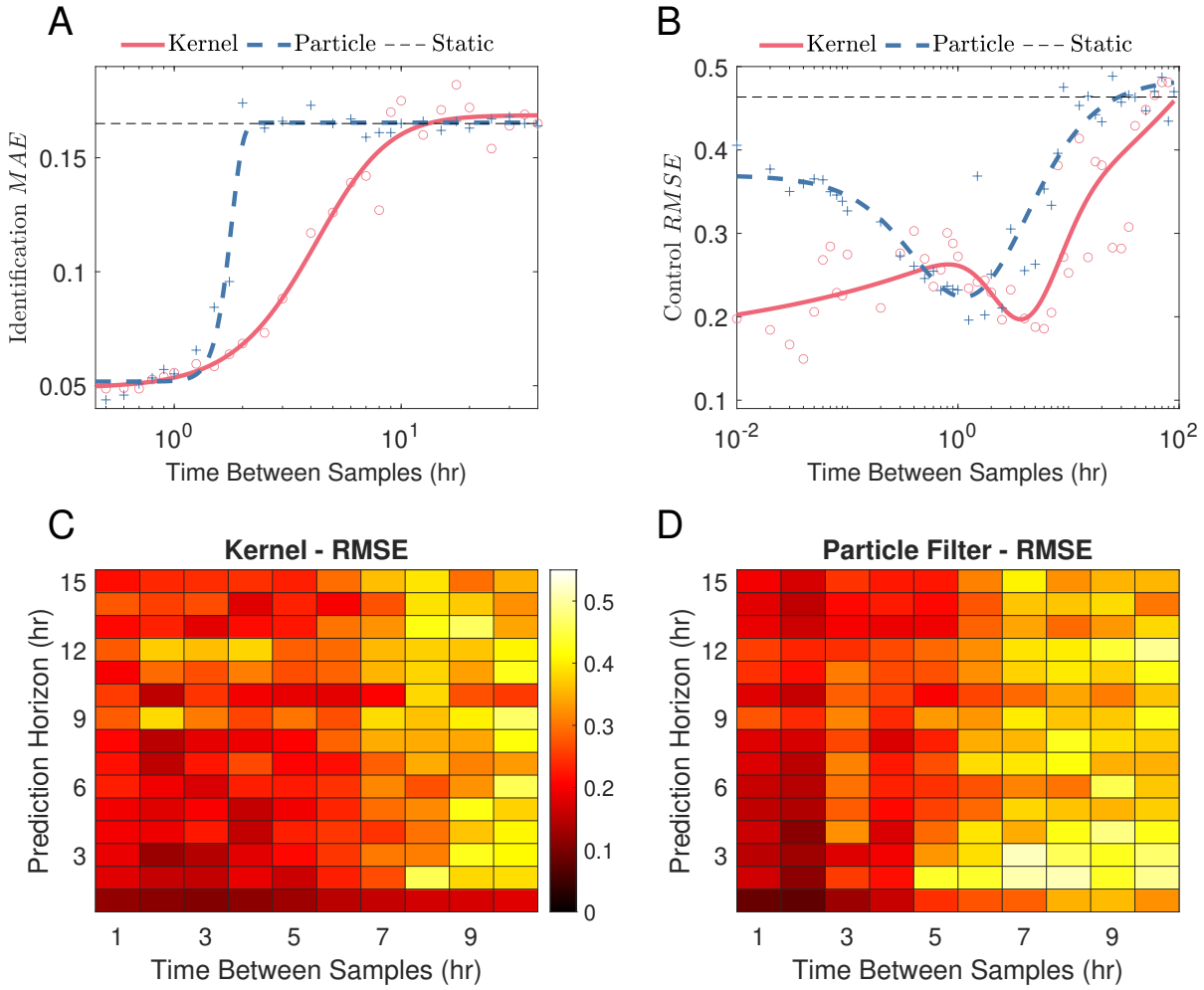

Figure 9: A) The identification Mean Absolute Error (*MAE*, Eq.(7)) of the two filters is compared over a range of sampling rates for the P53 oscillator. The simulation parameters for the kernel and particle filters are noted in Fig. 7. B) The control Root Mean Squared Error (*RMSE*, Eq.(9)) of the two filters are compared over a range of sampling rates for the P53 oscillator, with simulation parameters as in Fig. 8. C) and D) The control *RMSE* of the kernel filter (C) and particle filter (D), compared over a range of sampling rates and prediction horizons, N, for the P53 oscillator. Other simulation parameters as in Fig. 8.

4 Conclusion

In this work, we demonstrated that kernel filters are valid for fitting model parameters according to online measurements at low sampling rates through simulations of two genetic oscillators. Particle filters are preferable for some plant systems at faster sampling rates, yet at reduced sampling rates, a kernel filter should be used (sampling less than once a minute for the synthetic gene network and less than an hour for the P53 system). Fitted mathematical models can then be added to a model-based controller for a significant increase in controller performance at lower sampling rates.

The performance of the kernel filter could be improved in several ways. First, gradient-decent-based optimisers settle on local minima within the landscape of the *HSIC* cost function. A grid search algorithm has been used here, increasing the computational burden significantly with the number of parameters in the model and limiting the selected parameter values to defined discrete groups. A nonlinear solver that could find an acceptable minimum would drastically reduce the computational burden of adapting models with more parameters. The kernel filter could also use Bayesian inference to find the optimal parameter set in the feature space (rather than the particle filter that uses Bayesian inference in the problem space).

Secondly, the controller currently uses an exact model of the system (incorrect parameter values, but the model structure can exactly replicate the observed system). In real applications, fitting a simplified and reduced order model might be beneficial [43]. We hope our results will facilitate the implementation of cybergenetics for multiple applications across biological systems.

Author Contributions

B.S. designed and carried out the simulations in Matlab2024a and wrote the manuscript with inputs from all the authors. L.M. and L.R. conceived the project, contributed to manuscript writing and supervised the entire work.

Acknowledgement

An EPSRC DTP Scholarship supports B.S. L.M. is supported by an EPSRC fellowship (grant EP/S01876X/1) and a BBSRC transition award (grant BB/W013959/1).

Competing Interests

All authors declare no financial or non-financial competing interests.

Data Availability

Data sharing is not applicable to this article as no datasets were generated or analysed during the current study.

Code Availability

The underlying code for this study can be accessed via this link https://github.com/Ben-Smart/Kernel_filter_base_Adaptive_Controller.

Supplementary Information

The Supplementary Material for this article can be found online at: ...

References

[1] D. Del Vecchio et al. Control theory meets synthetic biology. *Journal of The Royal Society Interface*, 13(120), 7 2016.

[2] M. Khammash et al. Cybergenetics: Theory and Methods for Genetic Control System. *Proceedings of the IEEE Conference on Decision and Control*, 2019-December:916–926, 12 2019.

[3] Gianpio Caringella, Lucia Bandiera, and Filippo Menolascina. Recent advances, opportunities and challenges in cybergenetic identification and control of biomolecular networks. *Current Opinion in Biotechnology*, 80:102893, 4 2023.

[4] Benjamin Smart, Irene de Cesare, Ludovic Renson, and Lucia Marucci. Model predictive control of cancer cellular dynamics: a new strategy for therapy design. *Frontiers in Control Engineering*, 0:17, 9 2022.

[5] Alex Simpkins. System Identification: Theory for the User, 2nd Edition (Ljung, L.; 1999) [On the Shelf]. *IEEE Robotics & Automation Magazine*, 19(2):95–96, 6 2012.

[6] Jerker Nordh, Torbjörn Wigren, Thomas B. Schön, and Bo Bernhardsson. Particle filtering based identification for autonomous nonlinear ODE models. *IFAC-PapersOnLine*, 48(28):415–420, 1 2015.

[7] Xin Liu and Mahesan Niranjan. State and parameter estimation of the heat shock response system using Kalman and particle filters. *Bioinformatics*, 28(11):1501–1507, 6 2012.

[8] By Erdal and J Associate Member. Adaptive Modeling, Identification, and control of dynamic structural Systems. I: Theory. *Journal of Engineering Mechanics*, 115(11):2386–2405, 11 1989.

[9] Mark K. Transtrum, Benjamin B. Machta, Kevin S. Brown, Bryan C. Daniels, Christopher R. Myers, and James P. Sethna. Perspective: Sloppiness and emergent theories in physics, biology, and beyond. *The Journal of Chemical Physics*, 143(1):010901, 7 2015.

[10] L. Marucci. Nanog Dynamics in Mouse Embryonic Stem Cells: Results from Systems Biology Approaches. *Stem cells international*, 2017, 2017.

[11] Mouhacine Benosman. Model-based vs data-driven adaptive control: An overview. *International Journal of Adaptive Control and Signal Processing*, 32(5):753–776, 5 2018.

[12] Itzhak Barkana. Simple adaptive control - a stable direct model reference adaptive control methodology - brief survey. *International Journal of Adaptive Control and Signal Processing*, 28(7-8):567–603, 7 2014.

[13] I. De Cesare et al. ChipSeg: An Automatic Tool to Segment Bacterial and Mammalian Cells Cultured in Microfluidic Devices. *ACS Omega*, 6(4):2473–2476, 2 2021.

[14] E. Pedone et al. A tunable dual-input system for on-demand dynamic gene expression regulation. *Nature communications*, 2019.

[15] L. Postiglione et al. Regulation of Gene Expression and Signaling Pathway Activity in Mammalian Cells by Automated Microfluidics Feedback Control. *ACS Synthetic Biology*, 7(11):2558–2565, 11 2018.

[16] E. Pedone et al. Cheetah: A Computational Toolkit for Cybergenetic Control. *ACS Synthetic Biology*, 10(5):979–989, 5 2021.

[17] B. Shannon et al. In Vivo Feedback Control of an Antithetic Molecular-Titration Motif in Escherichia coli Using Microfluidics. *ACS Synthetic Biology*, 9(10):2617–2624, 10 2020.

[18] F. Menolascina et al. In-Vivo Real-Time Control of Protein Expression from Endogenous and Synthetic Gene Networks. *PLoS Comput Biol*, 10(5):1003625, 2014.

[19] M. Khazim et al. Towards automated control of embryonic stem cell pluripotency. *IFAC-PapersOnLine*, 52(26):82–87, 1 2019.

[20] Andreas Milias-Argeitis, Marc Rullan, Stephanie K. Aoki, Peter Buchmann, and Mustafa Khammash. Automated optogenetic feedback control for precise and robust regulation of gene expression and cell growth. *Nature Communications 2016 7:1*, 7(1):1–11, 8 2016.

[21] Nisha Ramesh and Tolga Tasdizen. CELL TRACKING USING PARTICLE FILTERS WITH IMPLICIT CONVEX SHAPE MODEL IN 4D CONFOCAL MICROSCOPY IMAGES. *Proceedings / ICIP ... International Conference on Image Processing*, 2014:446, 1 2014.

[22] M. N. Nounou, H. N. Nounou, and M. Mansouri. Model-based and model-free filtering of genomic data. *Network Modeling and Analysis in Health Informatics and Bioinformatics*, 2(3):109–121, 3 2013.

[23] Zhou Fang, Ankit Gupta, and Mustafa Khammash. Convergence of regularized particle filters for stochastic reaction networks. 10 2021.

[24] Alexander Medvedev, Zhanybai T. Zhusubaliyev, Olov Rosén, and Margarida M. Silva. Oscillations-free PID control of anesthetic drug delivery in neuromuscular blockade. *Computer Methods and Programs in Biomedicine*, 171:119–131, 4 2019.

[25] Ofir Lindenbaum, Amir Sagiv, Gal Mishne, and Ronen Talmon. Kernel-based parameter estimation of dynamical systems with unknown observation functions. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(4):043118, 4 2021.

[26] P. Rapp. Why are so many biological systems periodic? *Progress in Neurobiology*, 29(3):261–273, 1 1987.

[27] J. Hasty et al. Synthetic Gene Network for Entraining and Amplifying Cellular Oscillations. *Physical Review Letters*, 88(14):148101, 3 2002.

[28] Naama Geva-Zatorsky, Nitzan Rosenfeld, Shalev Itzkovitz, Ron Milo, Alex Sigal, Erez Dekel, Talia Yarnitzky, Yuvalal Liron, Paz Polak, Galit Lahav, and Uri Alon. Oscillations and variability in the p53 system. *Molecular Systems Biology*, 2(1):2006.0033, 1 2006.

[29] J. C. (Joseé C.) Priéncipe, Weifeng. Liu, and Simon S. Haykin. Kernel adaptive filtering : a comprehensive introduction. page 209, 2010.

[30] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring Statistical Dependence with Hilbert-Schmidt Norms. pages 63–77. 2005.

[31] Carl Edward Rasmussen and Christopher K. I. Williams. Gaussian Processes for Machine Learning. *Gaussian Processes for Machine Learning*, 12 2018.

[32] NOMAD - A Blackbox Optimization Software : GERAD.

[33] Simo Särkkä. Bayesian Filtering and Smoothing. *Bayesian Filtering and Smoothing*, pages 1–232, 1 2013.

[34] Furkan Elmaz, Reinout Eyckerman, Wim Casteels, Steven Latré, and Peter Hellinckx. CNN-LSTM architecture for predictive indoor temperature modeling. *Building and Environment*, 206:108327, 12 2021.

[35] J. Rawlings et al. *Model predictive control : theory, computation, and design*. Nob Hill Publishing, 2nd edition, 2020.

[36] E. Chatzi et al. The unscented Kalman filter and particle filter methods for nonlinear structural system identification with non-collocated heterogeneous sensing. *Structural Control and Health Monitoring*, 16(1):99–123, 2 2009.

[37] Joseph H.A. Guillaume, John D. Jakeman, Stefano Marsili-Libelli, Michael Asher, Philip Brunner, B. Croke, Mary C. Hill, Anthony J. Jakeman, Karel J. Keesman, S. Razavi, and Johannes D. Stigter. Introductory overview of identifiability analysis: A guide to evaluating whether you have the right type of data for your modeling purpose. *Environmental Modelling & Software*, 119:418–432, 9 2019.

[38] Dennis Bray. Protein molecules as computational elements in living cells. *Nature 1995 376:6538*, 376(6538):307–312, 7 1995.

[39] H. Burak Gunay, Jayson Bursill, Brent Huchuk, William O'Brien, and Ian Beausoleil-Morrison. Shortest-prediction-horizon model-based predictive control for individual offices. *Building and Environment*, 82:408–419, 12 2014.

[40] Muhammad Rizwan Azam, Vadim I. Utkin, Ali Arshad Uppal, and Aamer Iqbal Bhatti. Sliding mode controller–observer pair for p53 pathway. *IET Systems Biology*, 13(4):204–211, 8 2019.

[41] Genki Kawamura, Mitsuru Hattori, Ken Takamatsu, Teruyo Tsukada, Yasuharu Ninomiya, Ivor Benjamin, Paolo Sassone-Corsi, Takeaki Ozawa, and Teruya Tamaru. Cooperative interaction among BMAL1, HSF1, and p53 protects mammalian cells from UV stress. *Communications Biology 2018 1:1*, 1(1):1–13, 11 2018.

[42] S. Perwez Hussain and Cutis C. Harris. p53 Biological Network: At the Crossroads of the Cellular-Stress Response Pathway and Molecular Carcinogenesis. *Journal of Nippon Medical School*, 73(2):54–64, 2006.

[43] L. Marucci et al. Derivation, identification and validation of a computational model of a novel synthetic regulatory network in yeast. *Journal of mathematical biology*, 62(5):685–706, 5 2011.