NSLS II TECHNICAL NOTE BROOKHAVEN NATIONAL LABORATORY	NUMBER 0053 Rev 1
AUTHOR Nathan Towne[1] and James Rose	DATE January 26, 2009
TITLE Design of a Linear-Quadratic-Gaussian Controller for a Single-Cavity Rigid-Bunch RF-System[2]	

Abstract

Linear quadratic Gaussian (LQG) controllers have the potential to reduce rf-system noise in synchrotron storage rings below what is ordinarily achieved with conventional proportional-integral (PI) regulators. But realizing this possibility requires careful design of the regulator to match the characteristics of the machine (which is simplified by top-off operation) and is computationally demanding. This study develops theoretical and computational tools for the construction of LQG regulators for discrete-time models of single-cavity rf systems coupled to rigid-bunch beams, and driven by rf amplifier noise. Matlab's control-systems toolbox (CST) is the primary computational tool, particularly for the steady-state Kalman estimator, the LQR feedback, and the LQG regulator. Of the rf signals available to the regulator, a subset is chosen that results in a stable, effective, and economical regulator. This floating-point LQG regulator is then analyzed to establish resolutions of state variables, ADCs, DACs, and matrix coefficients that, in a fixed-point regulator, provided essentially undiminished performance. A Simulink model computes beam noise in linear and nonlinear rf-system models with floating-point, fixed-point, and proportional-integral regulators. This machinery is applied to NSLS-II, CLS, and NSLS VUV rings showing bandwidth and amplifier noise suppression beyond what a PI regulator can do. Required signal resolutions are surprisingly moderate. Testing and tuning with response functions computed by Vlasov simulations are performed. Thoughts are given on the further validation and tuning of the linearized model by machine measurements, implementations in logic, and distributed architecture for multiple cavities.

[1] 1094 White Oak Lane, Farmington, NY 14425; towne56@uchicago.edu.

[2] Work performed in part under contract number 126615 for Brookhaven Science Associates, LLC.

1 Contents

2 Introduction

NSLS-II is to be a state of the art synchrotron light source with unprecedented brightness and transverse emittances [1]. Timing experiments demand the most stringent tolerance of beam phase and energy noise of any experimental technique. These tolerances for short (unstretched and compressed) bunches are a factor of ten or more below expected noise generated by klystrons driving the cavities, so a great deal of noise suppression with substantial bandwidths are required of the rf system. Landau cavities to lengthen bunches and extend lifetime is to be more commonly used by other experiments, but some other experiments using these stretched bunches still require a high degree of energy regulation to utilize the high spectral resolution of radiation emitted by insertion devices [1].

The primary noise issue in the rf system is rf noise from the klystrons due to power-supply ripple, which results primarily in phase noise. Feedback around the klystron is a way to suppress this noise before it becomes more deeply entrenched in the rest of the rf system, but klystron saturation is an issue that complicates such a solution [20]. A phase loop is a simpler solution capable of largely suppressing phase noise because it is largely decoupled from klystron saturation. But there is still residual amplitude (and phase) noise that exceed tolerances when only this loop is used.

A linear-quadratic-Gaussian (LQG) regulator [2, 3, 4] utilizes knowledge of the workings of its target system (the 'plant' to be regulated) to minimize in a least-squares sense an object function J consisting of the noise of critical internals determined, in the case of a light source, by the needs of the experiments, plus a quadratic measure of the cost of controlling the plant via the plant's inputs (rf drive), while the plant is driven by its noise sources. Comprehensive and accurate models of the dynamics of the system being controlled and the noise driving it are required to synthesize these regulators. While those needs can be onerous, these regulators can be astonishingly effective.

Rigid-bunch models of the beam are capable of accurately describing short bunches [5,14] in synchrotron storage rings. While this is not possible for fully stretched bunches, rigid-bunch models are still capable of describing partially stretched bunches with sufficient accuracy that LQG regulators are still effective, at the same time stretching provides sufficient lifetime improvement to meet the needs of the machine and its users. Thus LQG regulators can be applied to NSLS-II for partially stretched bunches as well as for unstretched or compressed bunches.

This study explores the use of state-space models and LQG regulators and their digital implementations in the NSLS-II storage ring. Because the use of LQG regulators can be comprehensive as easily as not, the function of a klystron phase loop is not treated separately, but is instead absorbed into the LQG regulator.

In an rf system, noise bandwidths and the complexity of the plant to be regulated by an LQG regulator are sufficiently large that a great deal of computing power is required to implement the regulator. A field-programmable gate array (FPGA) is needed to meet this need, particularly with fixed-point computation due to its economy in logic. In digital control of analog systems, noise issues arise from both the analog-to-digital (ADC) conversion at the controller input, and the digital-to analog conversion at the output of the controller. The objective function J of the LQG formalism determined by plant specifications and the cost function applied to the closed-loop system provides a means to evaluate the tolerable noise introduced by quantization, and hence the ADC and DAC resolutions on an input-by-input and output-by-output basis. Finite resolution of computations internal to the Kalman estimator, particularly to the kernel matrix computation, also introduces quantization noise that must be evaluated. These sources, too, can be evaluated and controlled by the use of the objective function applied to the closed-loop system where tolerable resolutions are imposed on results of the kernel calculation (the A matrix product) determining evolution of the state variables. In this way it can be assured that the regulator is capable of providing the performance needed for the task, without throwing unneeded bits that add to the cost, the heat load, the bit error rate in the computation and programming, and failure rate of the controller.

The kernel calculation is computationally intensive because the kernel (the A matrix) is a relatively large matrix. The dimension of the matrix -- the number of state variables -- is eight for each cavity plus twice the number of bunches. More than one bunch may be included to accommodate the possibility that the regulator can be used to suppress nearby coupled-bunch modes and coupled-bunch equilibrium-phase instabilities, which are a problem in large rings [6] because they are driven by the rf-cavity accelerating modes. So the number state variables and matrix elements to be computed is rather large. The good news on this front is that the kernel (and the regulator as a whole) may be transformed by a similarity transformation, one that leaves the kernel block diagonal with at most two-by-two blocks. In this form, the kernel is much less computationally intensive and may be computed serially with two multipliers. Although such a kernel transformation leaves the B and C matrices dense, those matrix products are to be computed via a multiply-accumulate architecture unaffected by the transformation.

A single-cavity digital architecture is developed in Sec. 5.2, one that serially computes the *A*, *B*, and *C* matrix products during the sampling interval. The sampling interval of the LQG regulator is to be in the one- to two-MHz range, while the low-level data rate of the controller is to be about 40 MHz [7]. Two multipliers/accumulators for the A matrix, and a multiplier/accumulator for each row of B (inputs), and each column of C (outputs) implement the computations serially with sufficient time over the LQG sampling interval to process each set of samples without additional parallelization. Prototype logic in Verilog [8] was developed and simulated to verify the logic design of the single-cavity regulator, although no effort was made to test in hardware for data-processing capacity, or to add pipelining for this purpose.

With multiple cavities, the number of state variables associated with cavities is multiplied, while the beam degrees of freedom remain the same. In a natural digital architecture for multiple cavities described in Sec. 7.4, each cavity has a digital controller and there is a system-level controller. The *A*, *B*, *C*, and *D* computations are distributed in a natural way among the cavity regulators and the system-level regulator, while the system controller has a further data-distribution function.

This entire endeavor requires having an accurate linear model of the actual rf/beam system within which the regulator is embedded. The scope of this study does not, unfortunately, include development and testing of LQG regulators with beam on real machines. As a substitute, synthesized regulators were tested against linear models derived from Vlasov-computed impulse-response functions. In this way, the potential performance of LQG regulators was assessed against independent models having a degree of realism.

Looking beyond Vlasov models of machines, accurate machine measurements are ultimately required from which plant models are accurately fit. It is suggested that each controller have embedded in its logic a frequency-domain network analyzer capable of exciting the rf system and synchronously measuring the system's response. This route provides assurance that system measurements use the same hardware as the regulators use to control the system. Such measurements would be undertaken periodically, perhaps daily, and would entail download of the updated coefficients to the controller(s).

Although a number of software packages exist designed for the development of LQG regulators, in this study Matlab [9] and its Control Systems Toolbox was used for the synthesis of the regulator, and Matlab's Simulink was used for simulating synthesized regulators and assessing their performance while acting on linearized models and with non-linear Simulink models. Please consult the documentation for Matlab and other Mathworks [10] products as needed while reading this report, particularly while reading Sec. 3. Also consult Hindi's notes [11] or other sources for mathematical background regarding linear-quadratic-Gaussian regulators, and Moroney's dissertation [3] or other source for a survey of quantization noise and more generally the synthesis of LQG regulators.

In the rest of this report is described the steps involved in the construction of a (floating-point) regulator including the rf-system model and its linearization into a Matlab LTI object (Sec. 3), how fixed-point regulators are constructed (Sec. 4), the mechanics of model tuning and validation via machine measurements and Vlasov response functions (Sec. 5), the application of these ideas to NSLS-II, Canadian Light Source (CLS) [17], and NSLS VUV rings (Sec. 6), and architecture and design of fixed-point regulators in logic (Sec. 7).

3 Floating-point LQG regulator

There are a number of steps involved in the construction of the floating-point regulator. The most basic are the construction of the linearized plant model, the Kalman estimator, LQR feedback, and the LQG regulator. But prior to these steps, processing delay must be incorporated into the model. Then, plant outputs available to the regulator must be chosen based on stability and performance, and a kernel transformation is used to simplify the kernel computation.

3.1 RF system linearized model

The rf system model is of a rigid-bunch beam in a single-cavity ring modeled in Simulink. In Figure 1, the `particles` blocks models the motion of point-like bunches in longitudinal phase space (τ, ε) under the influence of voltage kicks each revolution coming from the `RFMode` block (for the fundamental accelerating mode). The rf mode is driven by impulses from the rf amplifier and the bunches passing through it (q and tau inputs, respectively). The `Amplifier` block models the klystron, and the `coupling` block handles conversion of waveguide fields to equivalent charge impulses driving the cavity and outputs forward and reverse power intensities. RF feedback in the `RF feedback` block is there to suppress the reactive Robinson instability that is inherent in superconducting cavities driving rigid-bunch beams [12].

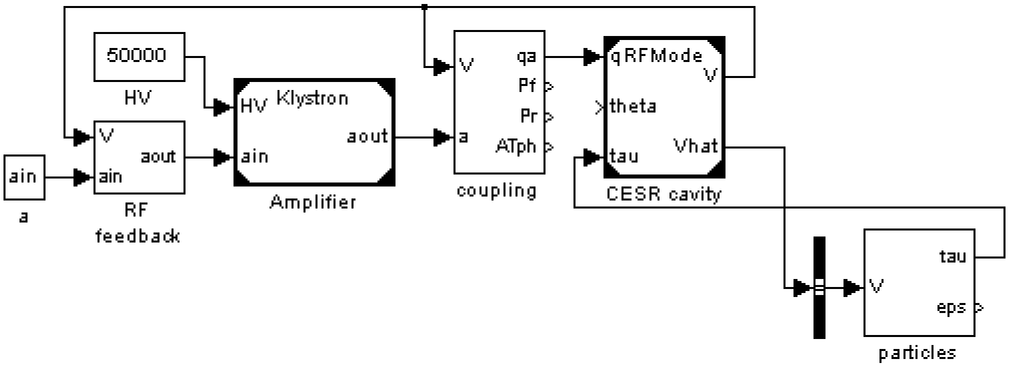

Figure 1: Basic single-cavity rigid-bunch mode in Simulink.

The synthesis of a regulator using CST requires starting with the model of Figure 1. The function `LTIBuild.m` handles the synthesis of the state-space model.

lti = LTIBuild('LTI5', 100000*Ts);

The first argument is a string that specifies the name of the Simulink model file, whose extension is .mdl. The second argument is the duration of simulation, after which is taken the operating point about which the model is linearized. `lti` is a state-space object returned by the function. It integrates the state-space matrix equations

$$\begin{aligned} x_{n+1} &= A\cdot x_n + B\cdot u_n + G\cdot w_n \\ y_n &= C\cdot x_n + D\cdot u_n + H\cdot w_n \end{aligned} \qquad (1)$$

The matrices are all computed by `LTIBuild` and are contained in `lti`. The vector *u* is to be used as a control input, the vector *w* is noise input, *x* contains the internal state variables, and *y* are outputs. The matrix *A* I term here the kernel. *B* and *G* are both contained in a single matrix `lti.B` = [*B*,*G*]; similarly for *D* and *H* contained in `lti.D` = [*D*,*H*]. The division between B and G in `lti`.*B* is for now one of interpretation determined by the goals of the model.

The basic model of Fig. 1, once linearized, is described by eight state variables. Additional state variables are required to accommodate shaping of the noise spectra and the signal-processing delay (discussed later).

There are a number of steps needed to prepare a model like Fig. 1 for linearization. To begin with, the amplifier block and CESR cavity block in Figure 1 are model blocks. Unfortunately, Matlab's linearization functions don't support model blocks, so the internals of these blocks must be copied to the model and model parameters replaced with values from the workspace.

`LTI5.mdl` of Figure 2 has these changes and also shows the addition of five input points and eight output points made available for linearization. Two of the input points are the I and Q phaser components of `ain` of Figure 1, two are I and Q phaser components of the noise injected at the klystron output, and the fifth is a field intensity applied to the input of the particles block. The I component of the input I/Q pair is aligned with the nominal phaser `ain` and normalized to `ain`. Similarly for the second I/Q pair at the output of the amplifier and the nominal phaser `a0` there. (`ain` and `a0` are computed when `NSLSII.m` is run. `NSLSII.m` and the files it calls define many machine parameters in the workspace.) The fifth input is meant to be a broad-band kicker such as a strip line. But I haven't done much with it because much too much rf power is needed there to make a difference.

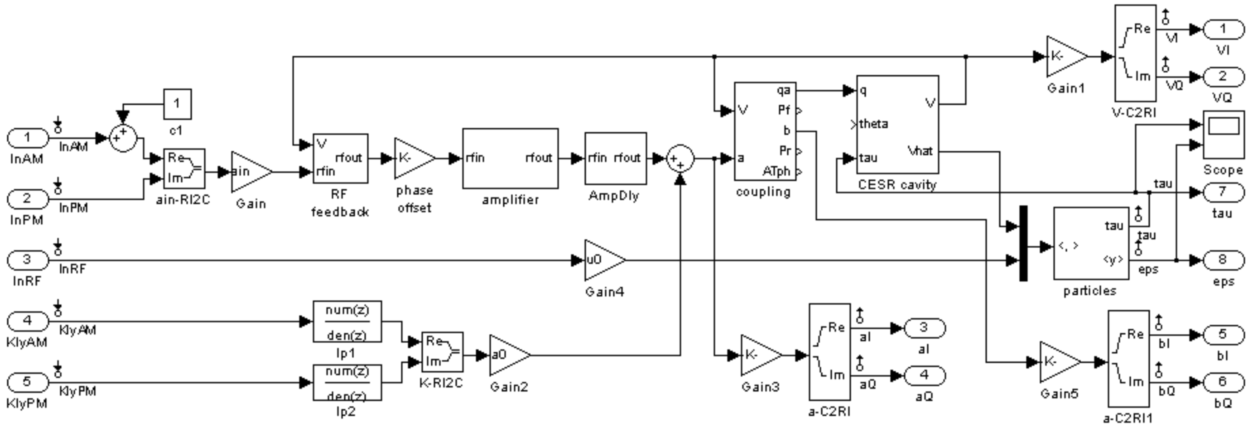

Figure 2: Full model used for linearization.

Each input and output point in a model to be linearized should be given a unique name. These names are used to specify the particular outputs to be taken as inputs by the regulator, and which inputs are to be supplied with signals by the outputs of the regulator. How this is done during construction of the regulator is described in Sec.3.5. The names are contained in cell arrays assigned to the properties `lti.inputnames` and `lti.outputnames` of `lti`. State-variable names are contained in the property `lti.statenames`. One refers to inputs, outputs, and state variables by their more user friendly names instead of by index. The names should be chosen so that the noise inputs are ordered after the signal (control) inputs.

Discrete-time models used for linearization should use the memory element instead of, e.g., the z^{--N} element. The latter block does not have the option to treat it as a delay for the purposes of linearization. When z^{--N} is used, results are all wrong.

The `particles` block of Figure 1 is able to handle an arbitrary number of bunches in a symmetric fill. Three bunches were for now chosen for the NSLS-II model so that one time step is a realistic number for the group delay in the amplifier chain, which is encapsulated in the `AmpDly` block of Figure 1. The correct number of delay elements must be added to the `particles` block of Figure 1 by hand.

3.2 Incorporating processing delay into the model

Processing delay is an integral part of the feedback loop. It must be built into the model even though it is physically part of the regulator. But adding memory elements to the inputs of the model generates an error when Matlab's `linearize` is run. To get around this problem, the `inputdelay` property of the signal inputs (but not the noise inputs) of the linearized model is set to 1, in effect adding a one-unit delay to the signal inputs. Then Matlab's `pade` function is run. That functions converts the input delays to poles on the *z* plane without errors.

A result of the addition of delays is that there are new state variables in the model, one for each input with a delay. (Minimizing the number of state variables is the reason to apply delays to only the signal inputs and not the noise inputs.) LTICovR gives the new state variables names.

3.3 The Kalman estimator

The Kalman filter estimates the plant state variables using a set of noisy inputs from the plant, given knowledge of the plant dynamics, statistical properties of the input noise *w*, and properties of the readout noise *v*. The particular outputs available to the estimator may be chosen at will, although the stability and performance of the LQG regulator depends on the choice. Input noise is specified by a covariance matrix $W = \langle u \cdot u^T \rangle$, where the angle brackets indicate a statistical average and *w* is white noise over the sampling bandwidth. In the model `LTI5`, the filters `lp1` and `lp2` roll off the white noise at 10 kHz to better simulate the klystron noise spectrum. The diagonal elements of *W* are chosen that they represent 1% amplitude noise and 3 degrees phase noise in 10-kHz bandwidth. Off diagonal elements are set to zero to simulate uncorrelated noise.

Similarly, the readout noise associated with the outputs y has covariance matrix $V = \langle v \cdot v^T \rangle$, where v is a column vector with the same dimensions as the outputs y. The estimator inputs see the noisy outputs $y_v = y + v$. Fairly small values of noise are chosen for V based on ADC data sheets, anticipating quiet ADCs.

From there, the Kalman estimator is generated by the `kalman` function.

3.4 LQG regulator and the closed-loop system model

Feedback from the estimator state variables back to the estimator and to the plant is determined by the matrix K minimizing $J = \langle x^T \cdot Q \cdot x \rangle + \langle u^T \cdot R \cdot u \rangle$, where user-supplied beam-phase and –energy noise tolerances define Q, and the cost associated with non-zero effort u defines R. The beam-based tolerances specify maximum rms jitter in τ and $\varepsilon = \delta E/E$; the reciprocal of the squares of these numbers are entered into the diagonal of Q with other elements being zero. Since there is a great deal of power available in the rf system, cost of the effort in u is small, and relatively small R is chosen accordingly.

From there, the Matlab (Control Systems Toolbox) function `dlqr` returns the feedback-matrix K given the matrices A, B (B for signal inputs, but not G for noise inputs), Q, and R.

The function `lqgreg` returns the LQG regulator given the estimator and the feedback matrix K. The regulator is a Matlab state-space object with its own A, B, C, and D matrices, as well as other properties.

The function `feedback` takes the plant and the regulator as inputs and returns the closed-loop system having the same inputs and outputs as the plant.

3.5 The choice of signal inputs

A problem with using LQG regulators is that they need not be stable. This does not mean that the regulator cannot control the plant, but that it is stabilized by feedback from the plant while it suppresses noise in the plant. The numerical experiments reported here show the best performance with unstable regulators, which tend to have higher gain, particularly when there is low open-loop group delay. But there are practical problems associated with unstable regulators. One is that when feedback is broken, the regulator state blows up. Another is that the high gain makes the closed-loop system (plant and regulator) more sensitive to variations of the behavior of the plant. To make the use of these regulators simpler for now, it is suggested that stable regulators be used initially, that a simple proportional-integral (PI) regulator be in parallel with the LQG regulator for use during injection, top off, etc., and having a digital architecture that can smoothly switch between them. Particularly during initial tests, digital logic can briefly switch from the PI regulator to an unstable LQG regulator; data buffers and logic can subsequently detect an unstable closed-loop system. This strategy may provide a future path to the use unstable regulators should their gains be found to be needed, effective, and robust.

So, for now, there is a need to find stable regulators. One strategy for doing this is to insert additional delay into the loop. Alternatively, a subset of plant outputs can be chosen that, when used to synthesize the regulator, result in stable regulators and acceptable closed-loop performance. Fortunately, with the eight available outputs of `LTI5`, there are typically a number of solutions, although the CLS model is an exception with only one useful solution.

One can test individual combinations using LTICovR with the syntax:

```
[ltinew sys kreg kregQ var] = LTICovR( …
      lti, …                      % the LTI object (the plant) from Ltibuild
      {'InAM', 'InPM'}, …         % cell array of plant control inputs (of the three in LTI5)
      {'KlyAM', 'KlyPM'}, …       % cell array of plant noise inputs (of the two in LTI5)
      {'VI', 'VQ', 'tau'}, …      % cell array of plant outputs (of the eight in LTI5)
      Input delay, …              % input delay in LQG samples
      Q, …                        % in $J = \langle x^T \cdot Q \cdot x\rangle + \langle u^T \cdot R \cdot u\rangle$
      R, …                        % in $J = \langle x^T \cdot Q \cdot x\rangle + \langle u^T \cdot R \cdot u\rangle$
      V, …                        % $V = \langle v \cdot v^T\rangle$ plant readbacks  noise covariance matrix
      W, …                        % $W = \langle w \cdot w^T\rangle$ plant noise inputs covariance matrix
      s, …                        % scale factor for quantization noise tolerance $Q_Q$ = s $Q$
```

```
        2 .^ -[10 7 5 0], …       % global A-, B-, C- and D-matrix quantization resolution
        false, …                  % boolean that specifies whether to plot regulator response functions
        true …                    % boolean that specifies whether to print performance diagnostics
);
```

Outputs:

- `ltinew` is `lti` stripped of unused signal and noise inputs,
- `sys` is the closed-loop system (with floating-point regulator),
- `kreg` is the floating-point regulator,
- `kregQ` is the fixed-point regulator, and
- `var` is an array containing $\langle x^T \cdot Q \cdot x \rangle$ and the maximum magnitude of the kernel eigenvalues.

To process all possible combinations of outputs for a given set of inputs searching for stable regulators, there is the function `bincomb` with the syntax:

```
bincomb(model_string, {'InAM' 'InPM'}, tol, inputdelay, chi2);
```

Its arguments are:

- `model_string` is a string specifying the LTI model to be linearized, typically ‘LTI5’;
- The second argument specifies plant inputs output by the regulator (one of three in `LTI5`);
- `tol` specifies a tolerance for $\langle x^T \cdot Q \cdot x \rangle$ below which results are printed;
- `Inputdelay` is the processing delay in time steps applied to the input of the model; and
- `chi2` is a sensitivity parameter used in the kernel scaling analysis discussed later.

For output combinations with $\langle x^T \cdot Q \cdot x \rangle$ below the tolerance, `bincomb` prints out $\langle x^T \cdot Q \cdot x \rangle$, the maximum eigenvalue magnitude, and the combination of outputs used by the regulator. `tol` = 1 returns a lot of hits with `LTI5`. They are clustered around 0.1. Beam noise at the user-supplied noise specification is $\langle x^T \cdot Q \cdot x \rangle = 2$.

3.6 The kernel in block diagonal form

From a computational perspective, there is quite a lot of computing to be done each sample time by the regulator, even with the sample time being nearly a microsecond. There is a need to avoid doing more computing than is necessary. Given any non-singular matrix S that acts on the vector of state variables, the regulator transforms as $A' = S \cdot A \cdot S^{-1}$, $B' = S \cdot B$, $C' = C \cdot S^{-1}$, and $D' = D$ without changing the external behavior of the regulator. The largest single block of computing is the $A \cdot x$ matrix multiply containing nearly 200 multiplies. A suitable transformation S can be used to greatly reduce the number of multiplies in $A \cdot x$. The function `LTIStateVarXForm` in Matlab code performs this transformation.

Since A is non-singular, it can be reduced to a diagonal matrix via a similarity transformation. But many or most of its eigenvalues are complex, which makes the similarity transformation and the transformed A complex valued. Because the original model is real, the eigenvalues are in conjugate pairs implying that A can be reduced to real block diagonal form where the complex arithmetic is done in real two-by-two blocks. Thus A can be reduced to block-diagonal form where the maximum block size is two. Matlab conveniently has the function `cdf2rdf` that performs this transformation.

The $A \cdot x$ product is reduced by this transformation to typically ~24 multiplies and the non-zero matrix elements of A of stable systems seem to fit neatly in the -1 to 1 scale.

The state variables are also reordered in `LTICovR` so that those with real-valued eigenvalues are grouped together at the end. This is done so that they can be paired, making the kernel uniformly block diagonal with block size two, anticipating the synthesis of fixed-point regulators, where the serial hardware performing the kernel computation is simplified. But the regulator must have an even number of state-variables for this to work.

4 Fixed-point LQG regulator

This section looks at the construction of a fixed-point regulator from a floating-point regulator. This involves:

- Quantization of the signal path, including quantization of state variables (Sec. 4.2), bit widths of the analog-to-digital conversions, and bit widths of digital-to-analog conversions (Sec. 4.3). Quantization of state variables requires assuring that quantization noise associated with state-variable resolutions are by some measure insignificant to the performance of the closed-loop system. Quantization noise associated with ADC and DAC resolutions must similarly be insignificant to the performance of the closed-loop system. Underlying these two topics are methods to ensure that matrix products, having multiplies and sums of several terms, preserve precision close to the level set by the data-path resolution (Sec. 4.1).
- The degree to which an LQG regulator, when represented by finite-precision matrix elements, has properties that remain sufficiently optimal that performance of the closed-loop system is degraded by a tolerable degree (Sec. 4.4). Resolutions required to meet this condition are determined by directly testing in numerical simulations.

The basic fixed-point regulator model is shown in Figure 3.

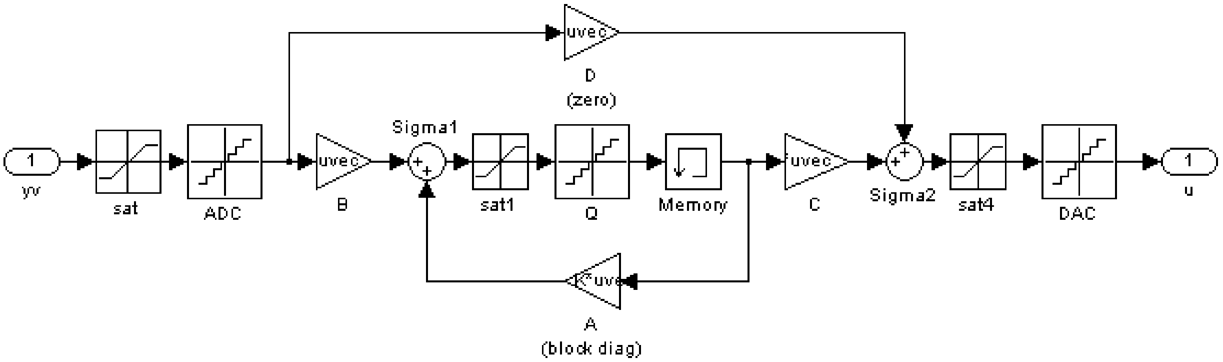

Figure 3: Fixed-point state-space model showing saturation and quantization blocks.

Sigma1 is referred to in this report as the kernel summation point.

4.1 Quantization noise and numerical precision

The truncation of numbers in a fixed-point processor results in errors arising from the result of a calculation being different from the exact calculation. This error is a sort of noise introduced by the truncation (modeled as) having a uniform distribution on the [-0.5, 0.5] interval (when rounding) of the least significant binary digit. This is the noise that one seeks to reduce by increasing the number of bits used to represent numbers in a fixed-point processor. It has variance $\sigma_0^2 = 1/12$.

When there are several addends summed together, all rounded to the same resolution, and each with noise σ_0, the noise associated with the individual terms are also summed (assuming noise arising from the truncations of the individual terms are uncorrelated). Thus the variance of the sum is $\sigma^2 = n\,\sigma_0^2$, where *n* is the number of terms in the sum. The sum has least significant bits that represent only noise. One then removes least significant bits by rounding so that all or most of the noise in the rounded sum is removed.

Looking ahead for the moment, the resolutions at the summation points of Figure 3 should at this point be regarded as fixed by the analyses of Secs. 4.2 and 4.3, which determine resolutions needed to meet the system noise specification. Since these points are all sums of products, there is spare resolution of addends coming from the products, which we can draw on to meet the needed resolution after each sum. These extra bits are here termed ‘extra accumulator bits’, or EABs, bits included in the sum but that are stripped away after the sum. We then ask, for a given number of EABs, how much noise remains in the rounded sum as a function of number of terms. Figure 4 answers this question in terms of effective number of bits (ENBs).

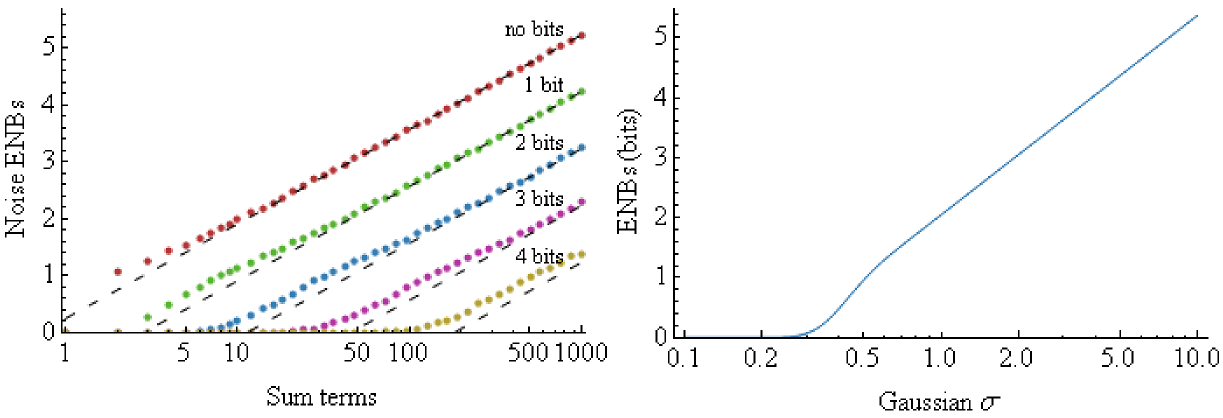

Figure 4: Quantization-noise effective number of bits (ENBs) vs number of terms summed for 0 through 4 extra accumulator bits (left) with rounding, and ENBs vs Gaussian σ (right) of quantized data. The left plot comes from a Monte Carlo calculation.

By ENB is meant the average number of bits of information carried by each numeric code: $ENB = \sum_i (-p_i)\log_2(p_i)$, where p_i is the frequency (probability) of the *i*-th numeric code. More generally, the sum may be over any symbol set carrying information over a communications channel. The Gaussian σ is related to the effective number of bits (ENBs) lost to quantization noise by Figure 4 right. For large σ, $ENB \sim \log_2(\sqrt{2\pi e}\sigma)$. Figure 4 left shows that, for large number of terms, there is roughly a bit of reduction of quantization noise with each extra accumulator bit. We add EABs to a particular sum until the residual noise is essentially zero.

Figure 4 left is computed for rounding. When truncation is used instead of rounding, there is oscillatory structure in the traces. Although this was not investigated, it is possible that the troughs may be used to advantage at the nose numbers of term. But the structure may be sensitive to deviations from white noise (time or other correlations). For this reason, in this report this structure is considered unwanted and necessitates the use of rounding in fixed-point arithmetic.

Figure 5 illustrates bit depths through the multiply-accumulate cycle of an inner product or a row of a matrix multiply. On the diagram, 'noise' refers to the accumulated quantization noise discussed in this section. The extra bits of resolution available for EABs are shown on the diagram as the difference in resolution between the product and the result.

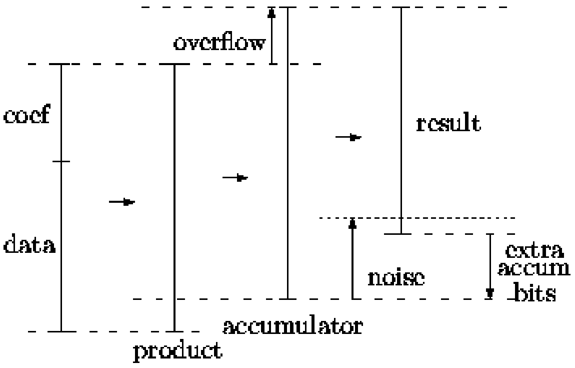

Figure 5: Bit depths while processing a row of a matrix product.

This diagram is integral to the discussions of the next two sections. In these sections,

- a sensitivity analysis determines the resolution of the ADC, which is one of the two 'data' of the kernel sum;
- a sensitivity analysis determines the resolution of the state variables, which are both a 'data' and the 'result' of the kernel sum (there are no overflow bits); and

- a sensitivity analysis determines the resolution of the DAC, which is the ‘result’ of the C and D matrix products.

From there the number of terms in each sum, and Figure 4, determine the EABs in each sum.

Of the allocated global budget χ_Q^2 for quantization noise carried through the closed-loop system to χ^2, each input, output, and state variable is allocated an identical $\chi_Q{}^2/n$, where n is the total number of inputs, outputs, and state variables.

4.2 Kernel scaling and state variable quantization

The basic idea of this topic is to determine and apply a kernel transformation that scales regulator state variables so that noise of variance due to state-variable quantization together results in a perturbation of the *closed-loop* (time-averaged) $\chi^2 = \langle x^T \cdot Q \cdot x\rangle$ of the *plant* that is below significance. By ‘below significance’ is meant a factor of one hundred or more below the target $\langle x^T \cdot Q \cdot x\rangle$ derived from user-supplied tolerances. Since the state variables are not necessarily physical, we take the scale of quantization in each case to be simply one. With such a transformation, the resolution of the fixed-point state variables may safely be set to one. In this analysis it is assumed that quantization noise of the state variables are all uncorrelated.

This calculation is performed by constructing the numerical model of Figure 6, which is a closed-loop model of the plant controlled by the floating-point regulator. In this model, the regulator has additional inputs: one into each summation point of each state variable in the regulator. Those new inputs are interpreted as noise inputs, where noise of known variance is applied one state variable at a time, and the *plant* $\langle x^T \cdot Q \cdot x\rangle$ computed. This results in a diagonal sensitivity matrix. In the scaled regulator (after a kernel transformation scaling the state variables) we want the diagonal elements of that sensitivity matrix to all be of the same value mentioned in the previous paragraph. This way, the quantization noise of no state variable predominates over the others. (For reasons not known to me, this step must be iterated. Consult the source code for details.)

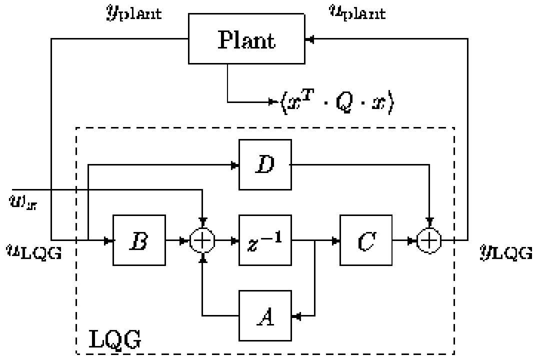

Figure 6: Kernel summation point sensitivity analysis.

This calculation is done by my function `LTIQNoiseState`. In it, the Matlab function `covar` is used to compute state-variable covariance matrices $X = \langle x \cdot x^T\rangle$ given a state-space model and noise input. X is used to compute the figure of merit $\chi^2 = \langle x^T \cdot Q \cdot x\rangle = \text{trace}(X \cdot Q)$.

This procedure establishes the resolution of the state variables, i.e., the smallest signals that matter. But it does not establish how large the state variables might become, i.e., the dynamic range. Simulations show the magnitude of the noise-driven signals present on the state variable. They presumably have a Gaussian distribution. One can assign a maximum magnitude as some factor times the Gaussian σ, e.g., 6σ, although this prescription may not allow the regulator to accommodate other than Gaussian noise sources, such as 1/f noise. Once the range is established, the correct number of bits can be assigned to the state variable.

As will be discussed later, I anticipate that the LQG regulator will be only be used during user time. Given this scheme, the regulator does not need to contend with injection, ramping, etc, although there are still drifts and glitches that are inevitably going to be present in the machine to contend with. So we cannot be definitive about the number of bits needed to represent state variables.

Closed-loop simulations of models of three machines with fixed-point regulators all show that some of the state variables are always zero, at least when excited by the noise model. This is interesting in that there is the possibility that some of the state variables may be dropped. Doing so eliminates those elements of the regulator matrix *A* as well as rows and columns of *B* and *C*, respectively, significantly reducing the regulator workload.

4.3 Analog-to-digital and digital-to-analog converter resolutions

In Sec.4.2, the impact of state-variable quantization noise was assessed by computing, based on the rf-system model, the impact of that noise on the figure of merit χ^2. In a similar way, we assess the impact of ADC (and DAC) quantization noise on χ^2 using analogs of the model of Figure 6, shown in Figure 7 and Figure 8 for ADCs and DACs, respectively. Following Sec. 4.2, we proceed by 1) applying known noise variances to the inputs (outputs) one at a time, 2) compute χ^2, and 3) scale the input (output) noise variance to meet the budgeted χ^2. In this way, ADC (and DAC) resolutions that provide the quantization noise variances that match the tolerable input (output) noise variances - for individual inputs (outputs) - are determined. The number of bits required of each input (output) then becomes the base-two logarithm of the required input (output) range (the ranges are of order +-1 in fractional units) divided by the resolution, with adjustment upward to an integer.

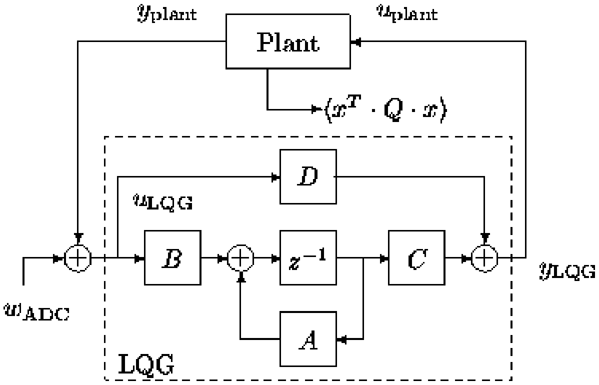

Figure 7: Model used to assess the closed-loop system's sensitivity to noise introduced by quantization in the ADC.

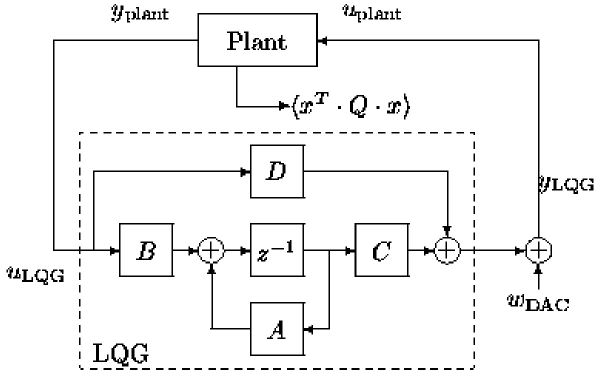

Figure 8: Model used to assess the closed-loop system's sensitivity to noise introduced by quantization in the DAC.

The calculation for the inputs is performed in Matlab by the function LTIQNoiseInput, which takes as input the plant LTI model, the floating-point regulator LTI model, a sensitivity matrix *Q*, and a print flag for diagnostics. The function returns the input noise variances, one for each input, that meet χ^2/n as determined by *Q*, where n is the total number of inputs, outputs, and state variables. LTIQNoiseInput again relies on the Matlab function covar to compute covariance matrices from the closed-loop model.

Paralleling the input calculation, the output calculation is performed in Matlab by the function LTIQNoiseOuput, which takes as input the plant LTI model, the floating-point regulator LTI model, a sensitivity matrix *Q*, and a print flag for diagnostics. The function returns the tolerable output noise variances, one for each output, that meet χ^2/n as determined by *Q*, where *n* is again the total number of inputs, outputs, and state variables.

Let us briefly review this strategy for the management of quantization noise in fixed-point regulators. Inputs, outputs, and state variables each contribute noise due to quantization. Assessing the impact of this noise on χ^2 as described in this section and Sec. 4.2 allows us to adjust resolutions so that none of the sources (inputs, outputs, and state variables) is significantly higher than the rest, and that all summed together do not break the noise budget imposed by the purposes of the machine. First, each of the three functions LTIQNoiseState, LTIQNoiseInput, and LTIQNoiseOutput by design impose a tolerable single-input, -output, and –state-variable χ^2 that is the tolerable system χ^2 divided by the total number n of inputs, outputs, and state variables, the factor n due to the summation noise sources (which are assumed uncorrelated). Second, one further provides a sensitivity matrix $Q = Q_Q$ to each of the three functions LTIQNoiseState, LTIQNoiseInput, and LTIQNoiseOutput that budgets to quantization noise a level of noise that is small compared to the system noise demanded by the purposes of the machine (a long way of saying that we add a few more bits to the quantized quantities). These bit counts are then applied to the precision scheme of Figure 5 for each input and output.

When applied to a model of the NSLS-II rf system with rigid bunches, to be discussed in more detail in Sec. 7.3, computed bit widths are in the range 10 to 12 bits for the ADCs, and 7 or 8 bits for the DACs when the target χ^2 is 1/100 the machine specification χ^2. So state-of-the-art ADCs and DACs are not required for this application.

Signal	**Tolerable variance**	**Minimum bits**
Inputs		
VI	2.8×10^{-7}	11
VQ	5.2×10^{-8}	12
tau	8.9×10^{-8}	11
Outputs		
InAM	6.6×10^{-5}	7
InPM	2.9×10^{-5}	7

This result can be further illustrated by computing (simulating) rms beam phase noise and beam energy noise as a function of ADC and DAC resolutions, all ADCs, and all DACs set to the same number in a fixed-point closed-loop model. Figure 9 shows the results of simulations for a non-linear NSLS-II model and a fixed-point regulator.

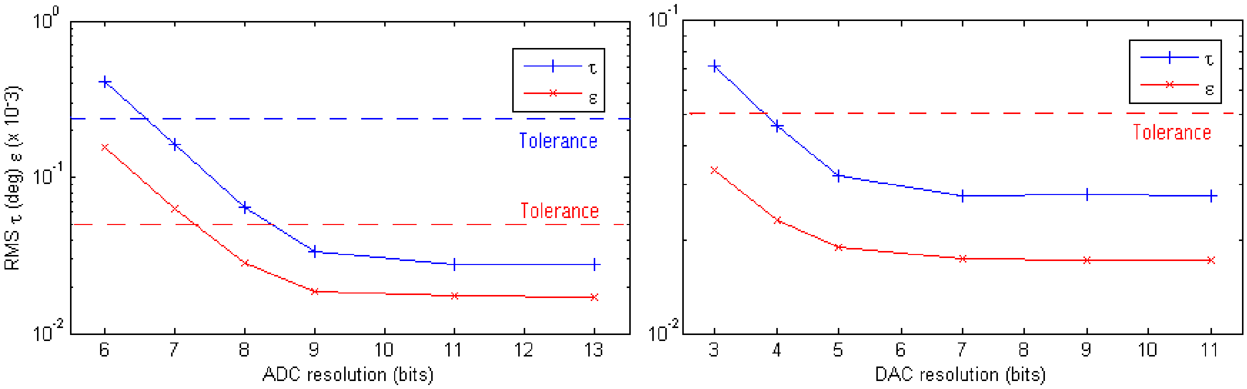

Figure 9: Simulated RMS noise vs. uniform ADC and DAC resolutions in a closed-loop NSLS-II model that is driven by amplifier noise of 1% amplitude and 3-degrees phase rolling off at 10-kHz frequency.

These results suggest that rather modest performance is demanded of the ADCs and DACs. These figures may be modest enough that there may be some skepticism over them. To allay this skepticism, it is instructive to compare the noise figures associated with the beam phase specification and compare them with those of high performance, high-speed ADCs. The 0.14 rf degree noise figure [1] corresponds to 0.24% of carrier intensity, $6\text{x}10^{-6}$ in terms of carrier power in the beam signal, and -52.2 dBc in something like 20-kHz bandwidth. Noise outside this bandwidth does not impact most experiments.

Contrast these figures with two Linear Technology (LTC) ADCs, the 14-bit LTC2249, and the 16-bit LTC2209. The former's noise is 0.021% of a sine-wave carrier rms, 4.6 $\text{x}10^{-8}$ of carrier power, -73.3 dBc, 11.6 ENBs, and 0.012 rf degree phase-noise equivalent over the full sampling bandwidth. The latter's noise is 0.014% of a sine-wave carrier rms, 2 $\text{x}10^{-8}$ in terms of carrier power, a remarkable -77.2 dBc, 12.3 ENBs, and 0.008 rf degree phase-noise equivalent over the full sampling bandwidth. These numbers are at high sample rates.

Figure 10 showing data taken at CLS [13] using H. Ma's digital controller [7] further illustrates the magnitudes of these numbers. It shows the cavity-signal spectrum from a large time-domain data set while controlling the rf system with 250-mA beam in the ring. There are many spurious spectral lines superposed on a continuum background, the latter with density at about ~ -118 dBc per Fourier channel. Total noise recorded by the controller in the +-50-kHz bandwidth is -68 dBc, or, in terms of phase noise, 0.023 rf degrees rms. So that test showed quite low total noise in the cavity and quite a low continuum noise floor of the signal. Yet ADC noise from the data sheets is quite far below the signal noise, and the theoretical quantization noise floor is less still. So these data illustrate that modern 14-bit ADCs do not limit the performance of a digital regulator in this application.

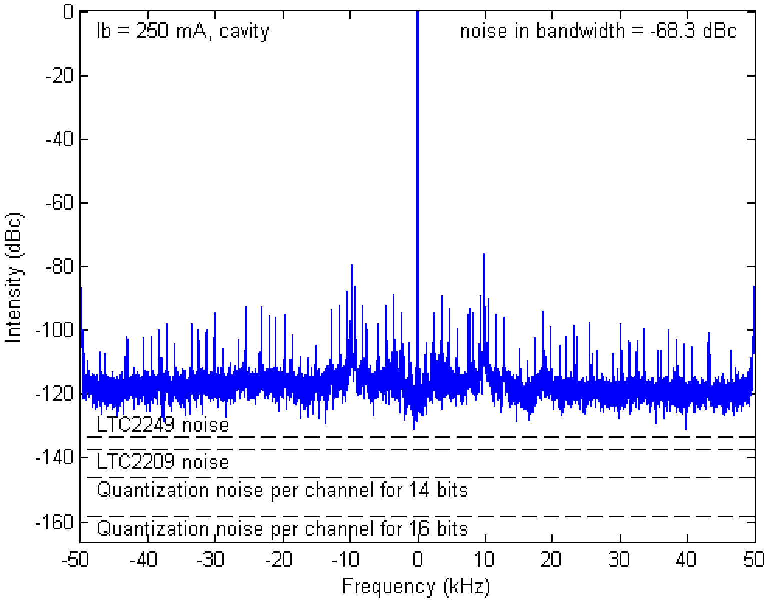

Figure 10: Spectrum of the cavity signal from data taken by H. Ma, J. Rose, and H. Song at CLS [13] using H. Ma's digital controller [7]. The data were taken with feedback provided by the controller, and with beam current at 250 mA. The dashed quantization noise per channel lines are for ideal 14- and 16-bit ADCs, while the dashed noise floors are from data sheets for LTC ADC chips LTC2249 (14 bit) and LTC2209 (16 bit). Total cavity noise within the +-50-kHz bandwidth is -68 dBc, equivalent to 0.04% rms amplitude, or 0.023 rf degree. Dashed lines take into account the controller's four-sample I/Q detection.

One can use of the fact that the I/Q data rate is higher than the LQG sample rate to suppress the noise in the data stream through a low-pass filter prior to decimation. Given that the I/Q rate is to be around 10 MHz and that the LQG rate is in the 1-2 MHz range, the oversampling factor is in the 5 to 10 range. This translates, for uncorrelated noise, into that factor of reduction of in-band noise variance, or one or two ENBs, plus some delay due to the averaging.

4.4 Matrix element quantization

The state-space *A*-, *B*-, and *C*-matrix quantization is established by direct word-length computation (ref. [3], Sec. 6.5), i.e., by adjusting the coefficient resolution of a matrix collectively and looking for changes in the closed-loop plant χ^2. The resolutions are set so that the incremental χ^2 in total is below significance in terms of tolerance. Including the sign bit, resolutions tend to be about 12 bits for *A*, 9 bits for *B*, and 7 bits for *C*. More specific values are given for NSLS-II, CLS, and NSLS VUV models in sections 6.1, 6.2, and 6.3, respectively.

Since the architecture of the controller allocates distinct multipliers to the rows of *B* and columns of *C* (to be discussed in Sec.7.1, Figure 24), one may alternatively assign distinct bit widths to the rows and columns of these matrices. Such an approach takes advantage of variations of significance among the inputs and outputs and may yield some economy of logic.

One further verifies that the *A*-matrix quantization has not significantly changed A's largest eigenvalue compared to its distance to the unit circle.

5 Model tuning

As was mentioned earlier, the success of LQG regulators is dependent on accurate linearized plant models from which these regulators are synthesized. Numerical rf-system models that have not been tuned to match a machine can still qualitatively resemble the behavior of short-bunch beams in a single-cavity rf system [14], but they are not sufficiently accurate quantitatively to serve as templates from which regulators are synthesized. It is not clear even with the numerical sensitivity experiments discussed earlier how sensitive an LQG regulator might be to quantitative details of the model. So I think it is essential to perform machine measurements using controller hardware with which to fine tune the model before contemplating tests of a controller in a machine.

Even with the model carefully tuned to the measurements, discrepancies are still expected between the two, particularly when the model fails to capture an elemental component of the machine's behavior. Examples are the existence of quadrupole and higher-order multipole modes bunches, and potential-well distortion due to broad-band impedances. To address this issue, a further test is to run the controller against a model fit to the full details of the measurements, i.e., a model that is faithful to the measurements. Collectively, the response functions are numerous and contain a great deal of structure; as such, many more state-variables are required to represent them than in the rigid-bunch model – no doubt too many to be used to construct a practical LQG regulator. Nevertheless, it can be used to verify that the closed-loop system is stable when floating-point and fixed-point regulators synthesized from simpler models are regulating the detailed plant model.

Section 5 outlines rf-system response measurements with beam and how to fine tune the linearized plant model to match machine or Vlasov response functions. Vlasov-computed response functions are introduced as a surrogate for machine measurements when such measurements are not available. Although they are a poor substitute for machine measurements, they still provide a model of the rf system and beam with considerable dynamical detail. As such, they provide an independent and detailed plant model with which to synthesize and test LQG regulators. The use of such models is discussed in Sec. 5.1 prior to the discussion of the measurement of machine response functions using controller hardware (ADCs and DACs) in Sec. 5.2.

Sections 6.1, 6.2, 6.3, and apply the methods described to this point to Vlasov models of NSLS-II, CLS, and NSLS VUV rings. Section 6.4 regulates the NSLS-II model with a proportional-integral controller for comparison. Section 6.5 discusses the problem of amplifier gain compression, how the plant model compensates for it (a gain parameter), how gain compression is problematic for direct rf feedback, and a digital solution to the last problem. Section 6.6 presents qualitative estimates of sensitivity of the closed-loop system to variations of gains and phase shifts within the plant model.

5.1 *Vlasov-computed response functions*

Vlasov-computed response function provide a detailed model of bunches coupled to a cavity (or cavities) as a surrogate for real machine measurements. Such a model is available through Vlasov simulations [15], which accommodate coupling of higher-order radial and multipole modes within the bunch (and potential-well distortion). Although far from a final test, this model provides a test of controllers synthesized from the model of Figure 2 for effects beyond the model of Figure 2: Full model used for linearization.. Success of such a test would be encouraging, and failure would cast doubt on the entire enterprise. Working through this test has been a useful exercise for working through the details of matching the model of Figure 2 to the Vlasov model. This is not wasted effort because the discrepancies found will speed matching Figure 2 against the real machine behavior when response measurements are available.

The idea here is to compute response functions of the Vlasov model, from the two rf inputs to the outputs that are to be used by the LQG regulator. These response functions are used in two ways: 1) Figure 2 is tuned to them, which is then used to generate an LQG regulator; and 2) they are accurately fit to rational functions for use as a plant model faithful to the Vlasov-computed behavior. The latter fit is used to represent the rf system in stability test,s where it is configured with feedback through the synthesized LQG regulator (Sec. 5.4).

Tuning Figure 2 to the Vlasov data was done by computing Figure 2's linear model in Matlab's `frd` representation, and minimizing a weighted square of the difference between Figure 2's and the Vlasov `frd` model's response functions. The fit is not particularly sensitive to the details of the weighting. Parameters varied are the real and imaginary parts of the rf feedback, and `InAM` gain parameter. The `InAM` gain parameter accounts for the degree of amplifier saturation. The gain parameters is also applied to the cavity-voltage feedback path because it also is subject to amplifier saturation. Later, gain parameters for all inputs and outputs will be needed with machine measurements to absorb details of the input and output hardware.

Results of the two-input and four-output fits are given in the Figure 11, which represents NSLS-II with four CESR cavities. `InAM` and `InPM` are the inputs in columns, `VI` and `VQ` outputs are represented in the left pair of columns, and the τ and ε outputs are represented in the right pair of columns. For each output there are separate magnitude and phase plots. Blue traces are the Vlasov `frd` model, and green traces are the tuned Figure 2.

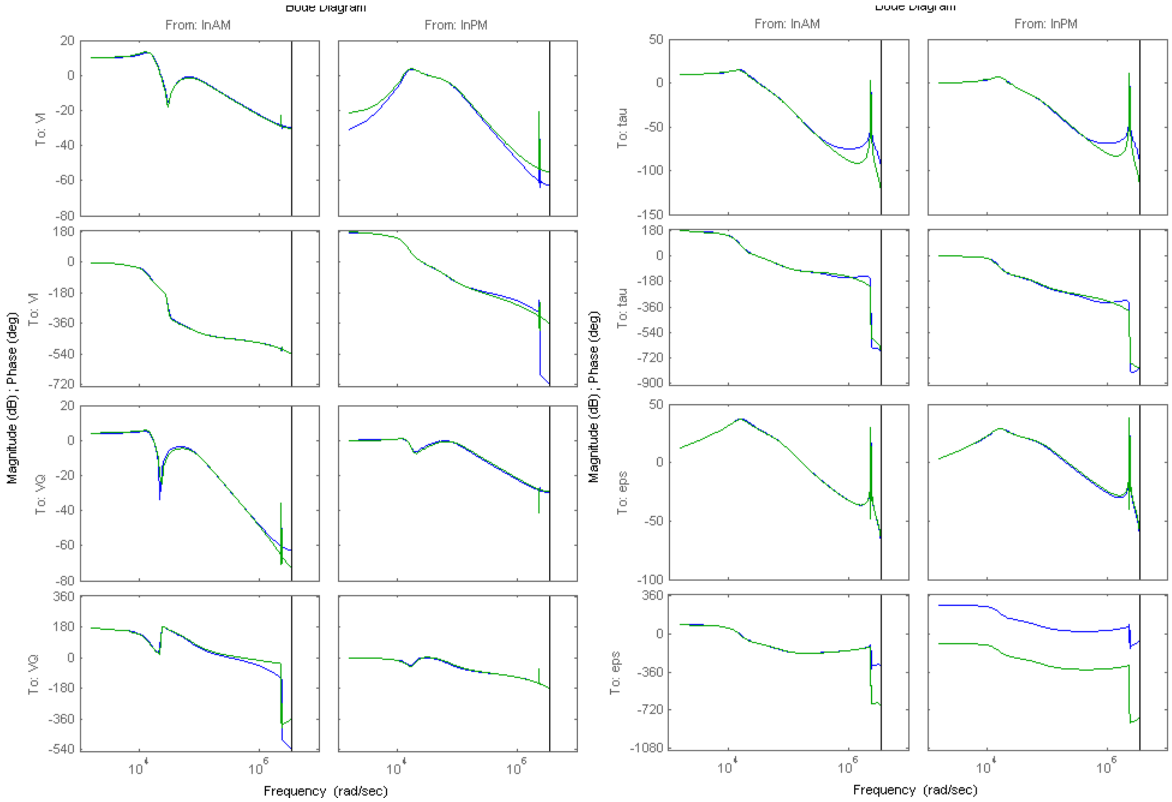

Figure 11: Response functions of the model of Figure 2 (blue) fit to Vlasov-calculated response functions (green). Input ports are `InAM` and `InPM` (left and right columns, respectively, of each pair of columns), and outputs shown here are `VI`, `VQ`, `tau`, and `eps` as labeled on the left edge of each pair of columns.

The resulting fits are remarkably good, although there are discrepancies. First, the `InPM` to `VI` response shows a discrepancy at low frequency, and at small amplitude. I do not regard this as significant. Second, there are differences is CB line shapes. But, since the 380-kHz offset is so large where the cavity impedance is greatly reduced, this also is not likely to be a problem. Third, and most significantly, the high-frequency behavior of the `InAM` to `VQ` response of Figure 2 is qualitatively wrong. A possible explanation is that there is an unaccounted for phase error that rotates the `InPM` to `VQ` response or the `InPM` to `VI` response into the `InAM` to `VQ` response, which is small at high frequencies. Beyond that I don't have an explanation for that behavior. Despite these problems, the overall quality of the fits confirms that the model of Figure 2 is capable of describing the behavior of my Vlasov model of short bunches is a single-cavity rf system with considerable precision.

Given that the amplifier in the model of Figure 2 has a time step of one LQG sample time, the ability to accommodate amplifier delay was added to the Vlasov code. The fits are sensitive to such details.

To generate the accurate analytic fits of the Vlasov response functions for use as a faithful Vlasov-simulated surrogate for machine data, Vlasov response functions are processed to generate rational-function fits as described in an earlier report [16], with the exception that Matlab's discrete-time `infreqz` is used instead of its continuous-time `infreqs`. This is done because Matlab cannot simulate an `frd` model consisting of numerical response data. Six state-variables per response function (denominator polynomials of degree six) are needed to model the data accurately to the degree shown in the following plots. Figure 12 shows eight (unlabeled) response functions for two inputs and four outputs (red) with their fits (blue), magnitude (left) and phase (right), showing the quality of the fits. The fits are exact nearly to the widths of the traces, with some distortion at the CB lines. That fits of stretched bunches required so many more state variables (12-15 for each fit) is no doubt due to mode coupling within the bunches.

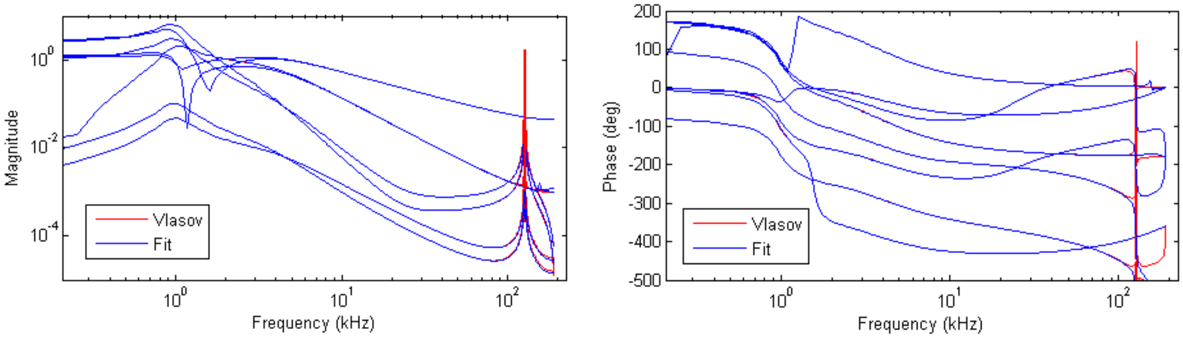

Figure 12: Rational-function fits to Vlasov-calculated response functions for use as a linear model of the beam/cavity system faithful to the calculated response functions.

Simulations showed that this Vlasov-derived rf-system model closed by the LQG regulator is stable. The model does not have the amplifier noise inputs of the model of Figure 2, so direct comparisons of noise suppression with Figure 2 closed by its LQG regulator are not possible. In any case, the success of this test is encouraging in that an economical LQG regulator derived from a different model (Figure 2) is stable with the model of Figure 12.

One may synthesize a controller directly from the fitted Vlasov model and be sure that the closed-loop system is stable, no doubt with good noise suppression. The problem with this idea is that each of the eight response functions is fitted independently. Consequently, each has an independent set of state variables resulting in a much more computationally intensive controller. But even so, a controller constructed in this manner may still be practical and used if needed.

5.2 Machine response measurements

It is suggested that machine response measurements are best done with the same DSP hardware that an LQG regulator would otherwise be using. The first reason is for this suggestion is that errors introduced by an alternate measurement apparatus, e.g., network analyzer, detector calibrations, cable lengths, etc., are avoided. A second reason is that the measurements can be integration with system operations so they providing more frequent, and timely data better and more simply able to track drifts. While performing the response measurements, a proportional-integral (PI) controller with low gain and long time constants can be regulating the system.

A problem with this idea is that feedback interferes with the measurement. The solution to this problem is to not only record the signals of interest, i.e., cavity field, beam signal, etc., while the system is being excited, but also to record the two drive signals (the ‘Source’ signals in Figure 13) output by the numerically controlled oscillator (NCO). With that information, the NCO–to-source response functions can be inverted and the true I and Q response functions between the source and the outputs de-embedded from the network. With careful work and low loop gain, the responses will disappear into the noise only at inconsequentially low and high frequencies, outside of frequencies at which the fits to Figure 2 are required.

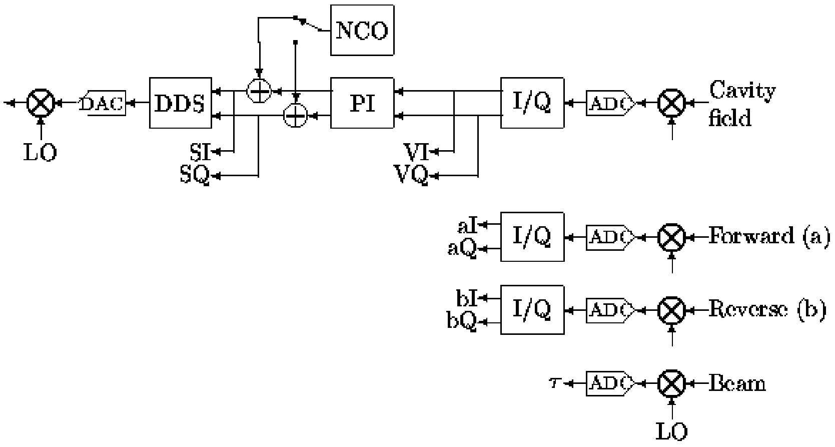

Figure 13: PI controller with network analyzer in a digital rf controller. The rf system (the amplifier, cavity, and beam) closes the loop between the rf output at the left, and the cavity field at the right. Set points are not shown in this figure. During normal operation, a digital regulator provides signals to the terminals of the direct digital synthesizer (DDS), and takes as inputs a subset of the seven I and Q signals from the cavity, forward, reverse, and beam. The I/Q blocks represent I/Q demodulation logic, and not NCO Fourier analysis of Figure 14.

In what follows, all quantities are in the frequency domain, i.e., each eight I and Q signals and τ is Fourier analyzed to a complex-valued function of NCO modulation frequency by an analyzer such as is shown in Figure 14. Furthermore, let S = $[\mathtt{SI\ \ SQ}]^T$ and V = $[\mathtt{VI\ \ VQ}]^T$ be two-component complex-valued vector-functions of NCO frequency. The value of these quantities depends on the modulation M_i applied by the NCO to the I and Q summation junctions, perhaps excitation applied to one and then the other. Like S and V, M_i is a two-component complex-valued vector-function of NCO frequency representing modulation type i. The rf system's response function G is a two-by-two response-matrix mapping rf modulation S to V. We have

$V = G \cdot S$

$S = PI \cdot V + M_i$

$\quad = PI \cdot G \cdot S + M_i$

Consequently,

$M_i = (1 - PI \cdot G) \cdot S$

Because S and other signals have two components, M_i can take on two linearly independent values in separate measurements. In other words, with the two measurements taken as a unit, $M = [M_1\ M_2]$ can be regarded as a square matrix. In a similar fashion, the vectors S, V, a, and b can be regarded as square two-by-two matrices, a column for each measurement with an independent modulation type. This means that, with square M known and square S and V measured, we have measured G:

$G = S^{-1} \cdot V$

Finally, the response functions to the other quantities are known.

$\mathrm{G}_a = S^{-1} \cdot a$

$\mathrm{G}_b = S^{-1} \cdot b$

$\mathrm{G}_\tau = S^{-1} \cdot \tau$

Note that τ and consequently G_t are only column vectors. Not shown is an equation for ε and G_ε. There are a total of 14 response functions to be measured this way, neglecting beam energy. At frequencies at which the PI controller has negligible response, we have $S = M$.

Returning to hardware, Fourier analysis can be done on chip. Figure 14 shows a straightforward Fourier-analyzer arrangement. After the NCO frequency is set and it is exciting the system, some dead time to permits let transients to damp out; then clear the registers and integrate for an integer number of NCO periods. Buffer or upload the I and Q outputs. Frequencies scanned should start from 30 Hz or so, and go to 100 kHz (higher if measuring the coupled-bunch lines). The I and Q outputs probably need quite a lot of bits (>16) to cover the range of signal intensities. The accumulator needs rather more bits on the least-significant end as described in Sec. 4.1 on numerical precision of fixed-point sums. To estimate this number, assume that the incoming I/Q data rate is 40 MHz/4 and the integration time is 10 ms, although this will vary with frequency. Then $n \sim 10^5$, and bits $\sim \log_2(\sqrt{2\pi e n \sigma_0{}^2}) \sim 8$ to 10.

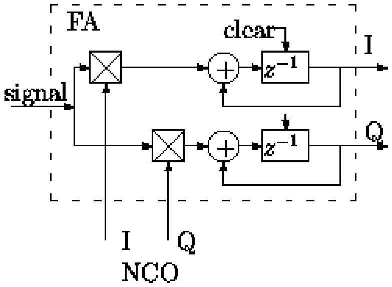

Figure 14: Fourier analyzer for a real-valued signal.

This procedure for the de-embedding of open-loop response functions from the closed-loop measurements may not be necessary. Recall that there is a need for feedback to suppress the reactive Robinson instability, and in particular, it must be active when performing response measurements. An alternate route for configuring the LQG regulator is to run it in parallel with the PI controller. In this configuration, the PI regulator is part of the environment in which the LQG regulator is embedded. This means that measurements of the linear properties of the system that the LQG regulator controls includes the running PI controller and uses the raw response measurements from *S* in Figure 13 to the various outputs (cavity field, beam, etc.) as the raw LTI object (the plant) from which the LQG regulator is synthesized. Thus the de-embedding procedure outlined earlier in this section is not used and the linear response measurements are substantially simplified. Furthermore, the integration of the LQG regulator with the larger rf controller is simplified in the sense that the PI controller is never frozen while the LQG regulator is active, but instead only the LQG regulator is stopped, reset, and started as needed. While a different model is needed to represent the plant in this configuration, it is not necessarily more complicated or more computationally intensive. Further study is needed to explore this configuration.

5.3 Model flow from measurements to logic

As has been discussed in some detail, two linearized rf-system models have been constructed from either machine response measurements or Vlasov-calculated response functions, one is a simplified model (the rigid-bunch model of Figure 2) used for the synthesis of the LQG regulator, and another that accurately reproduces measured or Vlasov-computed response functions and is later used for testing with the synthesized regulator. Figure 15 illustrates the flow of these models through the Matlab code and some of the code blocks used to synthesize or simulate these models. The file `NSLSII.m` oversees the process for the NSLS-II model and is the lowest-level code containing machine-specific data. Matlab code that synthesizes the LQG regulator is machine independent. The dashed arrow of Figure 15 indicates validation, that is, that acceptable performance of the closed-loop simulation of the faithful model with the synthesized LQG regulator suggests that the regulator may be useable in logic.

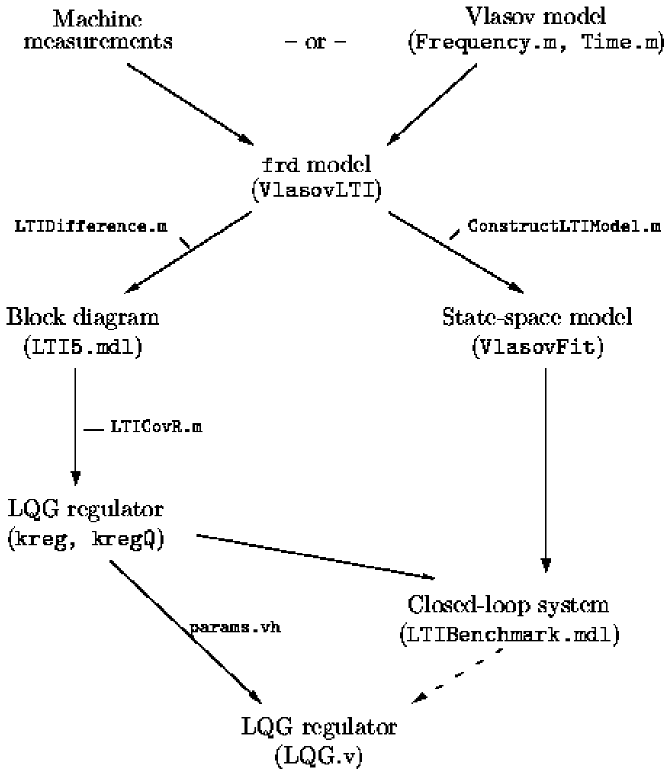

Figure 15: Model flow diagram showing model stages and their names in parentheses.

Objects and functions in this diagram are:

- `Time.m` – is generated by Mathematica code and contains Matlab code defining the `frd` object `VlasovLTI` housing impulse response functions. `VlasovLTI` is transiently used by `ConstructLTIModel.m` to build the state-space model `VlasovFit` accurately reproducing the measured or Vlasov-calculated response functions.
- `ConstructLTIModel.m` – constructs an accurate linear plant model from the measured or Vlasov-simulated frequency-domain response functions.
- `LTIDifference.m` – fits parameters of the non-linear rigid-bunch model of `LTI5.mdl` to the measured or Vlasov-computed response functions.
- `LTICovR.m` – Constructs the floating-point regulator `kreg` and the fixed-point regulator `kregQ`. It also assignes data to the workspace for use by `LTIBenchmark.mdl`.
- `LTIBenchmark.m` – performs closed-loop simulations of various plant models with various regulators (Sec. 5.4).
- `NSLSII.m` – NSLS-II-specific code handling the model flow of Figure 15. This is the machine-specific Matlab code layer.
- LQG.v – Prototype Verilog code defining the regulator in logic. Params.vh is an include file written by `LTICovR` defining some parameters used by LQG.v and lower-level Verilog blocks. More details are contained in Sec. 7.2.

5.4 Simulink model for open- and closed-loop simulations

In Secs. 3, 4, and 5.1, a number of linear and nonlinear models of the rf system and floating- and fixed-point LQG regulators were developed. The purpose of the regulators is to stably control and suppress noise in the rf-system models. With these rf-system and regulator models in hand, a means by which to test for stability and measure performance of the closed-loop systems is needed. There are two regulators to be tested: the floating-point and the fixed-point regulators. There are three related rf-system models: the non-linear model of Figure 2: Full model used for linearization. and its linearization fit to the Vlasov-calculated linear response functions, and the linear model accurately reproducing the Vlasov- and/or measured response functions. The Simulink model LTIBenchmark, part of which is shown in Figure 16, was developed for this purpose. The base non-linear model of LTI5, its linearization, floating- and fixed-point regulators generated by LTICovR, and a PI regulator are employed in LTIBenchmark. LTIBenchmark has a number of essential functions.

- First, it is needed to compare the closed-loop beam noise predicted by the Matlab function covar with those computed from time-domain simulations of linear and non-linear rf models and floating-point and fixed-point controllers. Thus it serves as a check of the synthesized regulators against the non-linear plant model, and against linear plant models derived from machine response measurements (Sec. 5.2) and/or synthetic (Vlasov-calculated, Sec. 5.1) response functions.
- Second, it is a tool used to hand tune the bit widths of quantized *A*, *B*, *C*, and *D* matrix elements of the controller. This is done by observing χ^2 (or beam phase and energy noise levels) of the closed-loop model consisting of a fixed-point controller and a plant model, and comparing it to either the theoretical calculated by covar, or the tolerance provided by *Q*.
- Third, it is used to compute state-variable rms intensities. Some have rather small intensities in the floating-point controllers, which end up always being zero in the fixed-point controllers. These state variables need not be part of the controller. The intensities also provide a means to estimate the range over which to represent the state variables (number of bits).

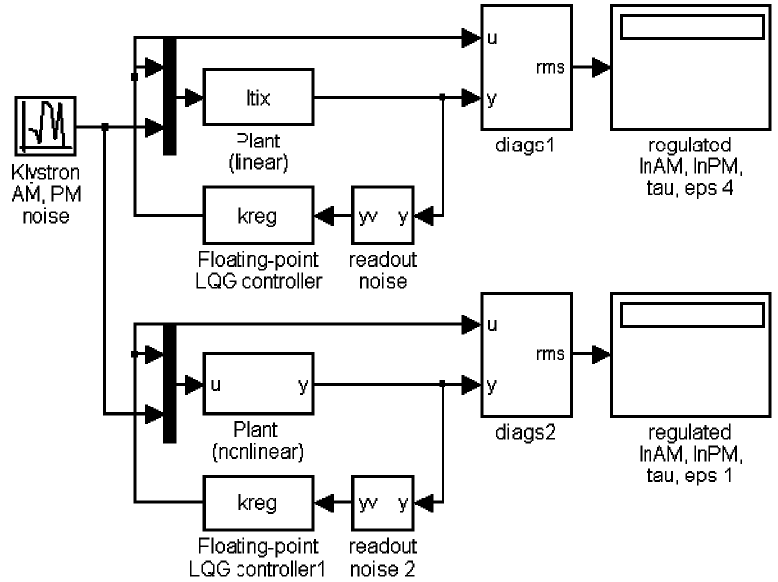

Figure 16: Part of the Simulink model LTIBenchmark for simulation of regulators with linear and non-linear plant models. Shown are linear and non-linear plant models regulated by a floating-point LQG regulator. Not shown are fixed-point regulators controlling plant models and an unregulated plant. The 'diags' blocks compute standard deviations, which are displayed in the right-most blocks, and route data to the workspace for later processing.

The non-linear plant model is equivalent to LTI5.

Models and parameters are taken from a number of variables in the base work space:

- ltix – the linearized model of the plant calculated by LTICovR.
- kreg – the floating-point controller calculated by LTICovR.
- kregQ – the fixed-point controller calculated by LTICovR.
- PlantOutputsUsed – routes the correct plant outputs to the input of the controller.
- taueps – indices selecting τ and ε from the plant output.

- The τ and ε outputs of the various simulations are routed to the work space and plotted in the frequency domain.

6 Performance estimates

6.1 NSLS-II model

When run on `bincomb`, the NSLS-II ring parameters returned a number of stable controllers. The one represented in tables 2 to 4 and Figure 17 uses VI, VQ, and tau (beam phase) as inputs. Amplifier noise is 1% amplitude and 3° RMS phase in 10-kHz bandwidth. Identical noise tolerances and readout noise were specified. Ring parameters are in `NSLSRing.m` and `NSLSII.m` and the model to linearize is `LTI5.mdl`.

Table 1

Plant	Controller	σ_τ (deg)	σ_ε $(\times 10^{-3})$
Linear	None	1.37	0.33
Linear	Floating point	0.017	0.015
Linear	Fixed point	0.019	0.015
Non-linear	Floating point	0.024	0.016
Non-linear	Fixed point	0.025	0.016

Table 2

Matrix	Resolution (bits)
A	12
B	9
C	6

Table 3

I/O	Resolution (bits)
ADC	11
DAC	7

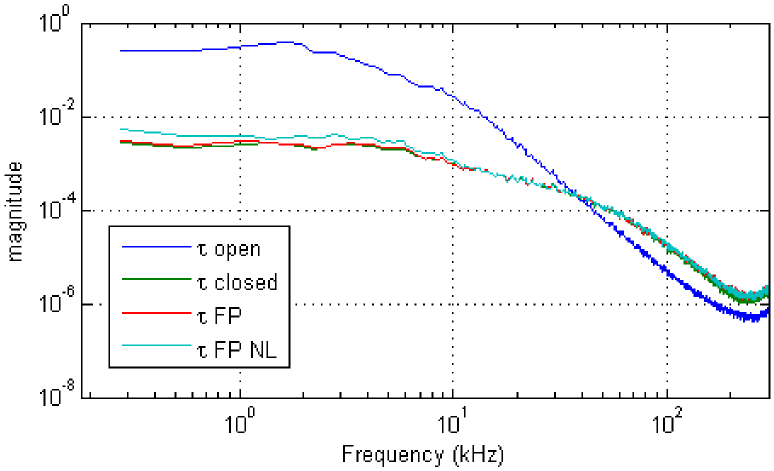

Figure 17: Spectrum of linearized model open loop (blue), coupled to a floating-point LQG regulator (green), and coupled to a fixed-point controller (red); non-linear model coupled to the fixed-point controller (aqua).

The exact shape of the frequency-domain response of the open-loop rf model is dependent upon the rf feedback, whose primary purpose is to suppress the reactive Robinson instability. The closed-loop response functions should not be sensitively dependent on the feedback, although I haven't experimented with this yet.

6.2 *Canadian Light Source model*

When run on `bincomb`, the CLS ring [17] parameters returned only a couple of stable controllers. The only useful one is represented in the tables 5 to 7 and Fig. 19 and uses VI, VQ, and tau as inputs. Amplifier noise is the same as in the other models: 1% amplitude and 3° RMS phase in 10-kHz bandwidth. Identical noise tolerances and readout noise were specified. Ring parameters are in `CLCRing.m` and `CLS.m,` and the model to linearize is `CLS1.mdl`.

Table 4

Plant	Controller	σ_τ (deg)	σ_ε $(\times 10^{-3})$
Linear	None	0.389	0.069
Linear	Floating point	0.023	0.0070
Linear	Fixed point	0.024	0.0072
Non-linear	Floating point	0.024	0.0071
Non-linear	Fixed point	0.026	0.0076

Table 5

Matrix	Resolution (bits)
A	9
B	6
C	5

Table 6

I/O	Resolution (bits)

ADC	11
DAC	9

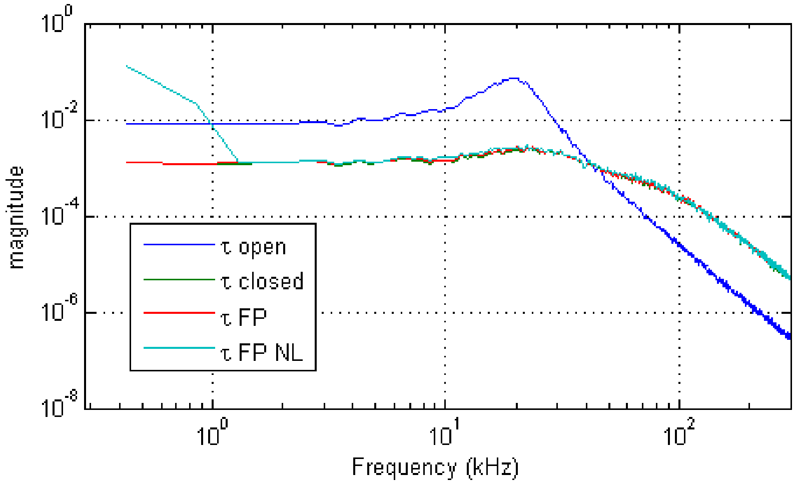

Figure 18: Spectrum of linearized model open loop (blue), coupled to a floating-point LQG regulator (green), and coupled to a fixed-point controller (red); non-linear model coupled to the fixed-point controller (aqua).

6.3 NSLS VUV model

When run on `bincomb`, the NSLS VUV ring parameters returned a number of stable controllers. The one represented in the following tables 8 to 10 and Fig. 20 uses VI, VQ, and tau as inputs. Amplifier noise is the same as in the other models: 1% amplitude and 3° phase RMS in 10 kHz bandwidth. Identical noise tolerances and readout noise were specified. Ring parameters are in VUV`Ring.m` and `VUV.m` and the model to linearize is `VUV1.mdl`.

Table 7

Plant	Controller	σ_τ (deg)	σ_ε $(\times 10^{-3})$
Linear	None	0.53	0.065
Linear	Floating point	0.0085	0.0023
Linear	Fixed point	0.0091	0.0030
Non-linear	Floating point	0.0091	0.0023
Non-linear	Fixed point	0.0096	0.0028

Table 8

Matrix	Resolution (bits)
A	12
B	8
C	6

Table 9

I/O	Resolution (bits)
ADC	11
DAC	7

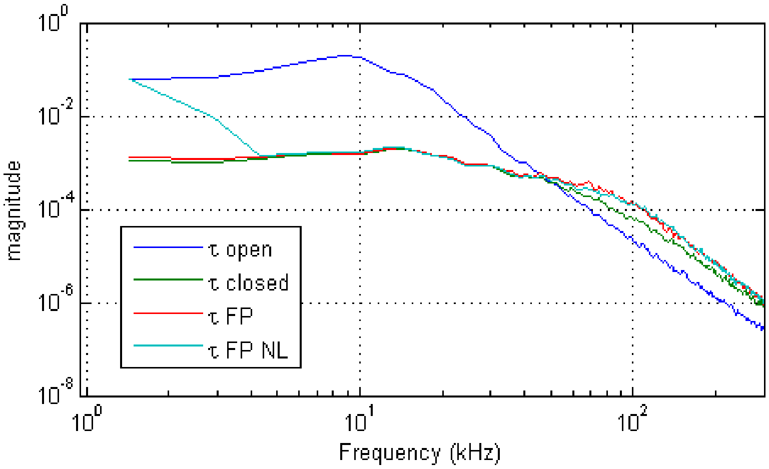

Figure 19: Spectrum of linearized model open loop (blue), coupled to a floating-point LQG regulator (green), and coupled to a fixed-point controller (red); and non-linear model coupled to the fixed-point controller (aqua).

6.4 Proportional-integral regulator in an NSLS-II model

It was suggested that I compare the performance of LQG regulators with a conventional proportional-integral (PI) regulator. Here I describe the results for the case of the NSLS-II machine.

LLRF systems typically sense the level and phase (or I and Q) of the cavity field and apply corrections back to the input of the rf amplifier (level and phase or I and Q). This feedback affects the stability of the beam, particularly if the feedback has significant bandwidth compared to the synchrotron frequency. One cannot do this by trying to suppress noise in the cavity to a high degree because this also suppresses perturbations of the cavity field by the beam. This reduces the impedance of the cavity, and consequently reduces damping of Robinson modes. Furthermore, rf feedback employed in the NSLS-II model to optimally damp Robinson modes and suppress the reactive Robinson instability would be spoiled by additional feedback, perhaps increasing amplifier noise transmitted to the beam. So one cannot, in a simple way, use high-gain feedback from the cavity field.

An approach to this problem I looked at is to consider what a single (scalar) loop could do to noise transmission to the beam. Because the cavity field is complex valued, as is the amplifier input, reference phasers on the complex plane much be chosen to get a real signal from the cavity field, and for a real input to modulate the rf. I intuitively chose to use AM modulation of the rf (parallel to the amplifier forward-wave-output phaser `a0`), and to sense the phaser component of the cavity field at the phase of `a0` times the cavity's (loaded) impedance, the cavity phase at which low-frequency perturbations of `a0` appear. This choice is stable and provides significant suppression of noise transmission to the beam in the NSLS-II model. The results are shown in Table 11 and Fig. 21 showing noise in the unregulated model and the regulated model with PI and LQG regulators. As with all the models of this report, the specified amplifier noise is 3° phase, and uncorrelated 1% amplitude Gaussian noise in 10-kHz bandwidth.

Table 10

Regulator	σ_τ (deg)	σ_ε $(\times 10^{-3})$
None	1.4	0.33
FP LQG	0.017	0.015
PI	0.40	0.15

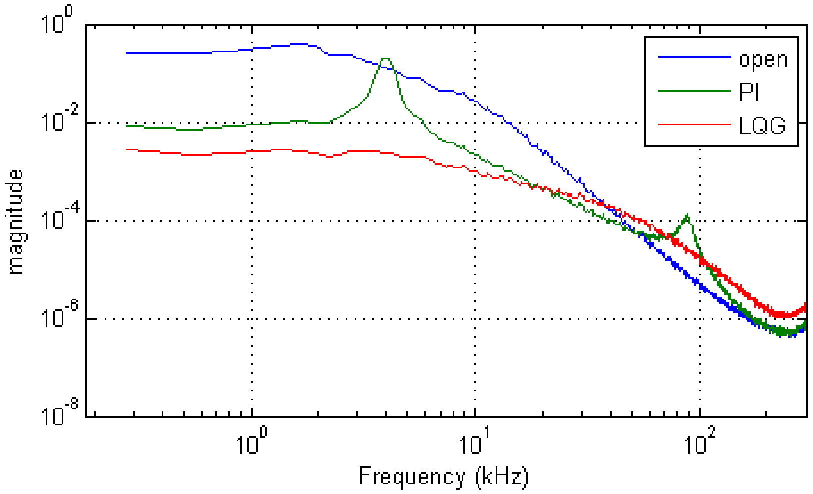

Figure 20: Noise suppression of the NSLS-II model with a proportional-integral regulator.

There is considerable suppression of the noise spectrum at lower frequencies. But, in terms of total RMS noise, the suppression is only by a factor of three for beam phase noise, and by a factor of two for energy over the unregulated model. Much of the remaining noise is due to a broad line at the synchrotron frequency. In this test, the gain has been pushed upwards to near instability. The peak at about 100 kHz is where phase margin becomes small and oscillations occur at slightly higher gain. The synchrotron line at 4 kHz may be stable or more stable in this case. The emergence of that line is due to the reduction of the impedance of the cavity by the feedback, which lessens damping of the Robinson mode by the cavity [14].

This particular feedback scheme did not, unfortunately, work at all with the CLS and VUV models. Only feedback with unity-gain bandwidth small compared to the synchrotron frequency was stable, and then with negligible rms noise suppression. Feedback with any bandwidth that results in stable beam is sensitive to rf feedback used and the resulting properties of the Robinson modes. I don't know in any detail why the NSLS-II model behaves so differently with PI feedback. But with so little bandwidth in the CLS and VUV models, the feedback is only able to correct for drifts. This is the behavior I had come to know working with the VUV ring. Any bandwidth at all, particularly with the phase loop, resulted in instability.

Another approach that is useful for suppressing amplifier noise is applying feedback around the amplifier to regulate gain and/or phase shift. Although amplifier noise is dominated by phase noise, the presence of some amplitude noise means that the use of a phase loop leaves residual noise even with high phase-loop gain. How much depends on the AM vs PM noise characteristics of the amplifier. Aside from this limitation, a phase loop is likely to be insensitive to amplifier saturation and could handle considerable bandwidth, and so could effectively suppress noise and be relatively easy to implement. Numerical experiments with a phase loop were successful at reducing the noise transmitted to the beam by nearly a factor of 100 or so, a factor similar to the performance of the LQG regulator. Because of the asymmetry of the AM vs PM noise in the model, the beam noise was sensitive to misalignments of the phase loop compared to the phase noise (a misalignment built into the klystron model). That factor of 100 requires a trim to correct for that misalignment.

An amplitude loop around the amplifier is sensitive to saturation, as any control scheme is.

All of these noise-suppression factors are limited by the delay in the loop. With lower delay, gains and bandwidths can be pushed higher and better noise suppression results.

6.5 *Amplifier saturation and gain compression*

Amplifier saturation is an inevitable problem. Although a closed-loop model did not show special sensitivity to amplifier saturation (Sec. 6.6), it does still present a problem for operation of these regulators deep in saturation, at the very least because the amplifier can simply run out of power. Even if the amplifier does not run out of power, the non-linearity of the amplifier output means that the system response functions drift and performance deviates optimal. A digital regulator can correct for this problem, either by a non-linear output that inverts the amplifier nonlinearity, or simply by a gain parameter that varies in a calibrated way with the operating point. Periodic amplifier response measurements as outlined in Sec. 5.2 in effect implement the latter.

But this method incompletely compensates for amplifier nonlinearity when, as in PEP-II [18],there is direct rf feedback in parallel with the digital regulator. This is because the digital gain parameter is not in the path of direct rf feedback. But the NSLS-II rf system uses direct rf feedback at low gain as a means to stabilize the reactive Robinson instability, unlike PEP-II, where it is used at high gain to suppress accelerating-mode impedance. That it is at low gain suggests that gain compression will not affect NSLS in the same way. But even if it does, rf feedback at low gain can easily be absorbed into the function of the digital regulator, perhaps into the PI regulator programmed to use the same measured gain parameter as the LQG regulator.

Measurement of the gain parameter requires a much simplified response measurement compared to the measurement outlined in Sec. 5.2. Referring to Figure 14, it requires measuring the ratio of the in-phase component of the amplifier forward amplitude `aI` to the in-phase component of the source, call it `sI`, at a single relatively low frequency and in a few tens of milliseconds. This can be done closed loop without the correction for the presence of the loop discussed in the section. The simplicity and speed of the measurement suggests that it could be run transparently during top off, although it likely is not needed every top off.

6.6 Sensitivity to variation of operating point and rf misalignments and mismatches

The behavior of a real machine will inevitably differ from the behavior of the linearized model upon which an LQG regulator is built. There can be a variety of reasons for this, including phaser misalignments, drifting components, and a discrepant operating point.

To get a general idea of the sensitivity of the closed-loop rf system to these sorts of changes, I introduced gain/attenuation and phase shifts in the amplifier chain at the points shown in Fig. 8, points 1 and 2 being phase shifts, and points 3 and 4 being gain/attenuation. Point 1 differs from point 2 in that it is outside the feedback loop, a relatively minor difference given that the rf feedback gain is only a few decibels. But they both have the effect of rotating the incoming signal phaser relative to the nominal.

A gain perturbation at point 3 would be expected to require compensation from the regulator, but would not have a significant impact on the amplifier operating point if the DC open-loop gain is large. In contrast, point 4 is downstream of the amplifier and would have a stronger impact on its operating point.

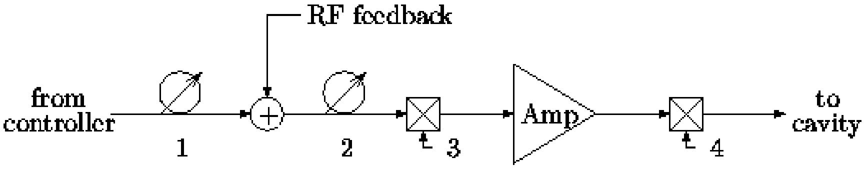

Figure 21: Four points in the amplifier drive chain at which perturbations are introduced.

I found that both of the phase perturbations could be over more than $\pm 40^\circ$ without instability and/or particle loss at both points 1 and 2, and with both floating-point and fixed-point regulators. Point 1 may have been slightly less sensitive that point 2 as would be expected with feedback, although the test was rough.

I also found that the loop is less sensitive to perturbations at point 3 than point 4 as expected. At point three, particles were not lost for gain perturbations in the 0.7 to 3 range, roughly. At point 4, the corresponding range is 0.93 to 4. So the loop is quite sensitive to amplifier saturation, as expected. The amplifier model exhibits strong saturation and level-dependent phase shifts. The operating point was intentionally chosen to be into saturation but with some output power to spare before it peaks and falls off at higher drive.

There is variation of beam noise with the perturbations, with the noise becoming quite large prior to beam loss.

Particles were lost perhaps due to the initial mismatch of the incoming phaser.

So this closed-loop model shows significant margin for perturbation of the plant, which is why early on I was encouraged by these results. Although these results are not very quantitative, it is my hope that model tuning, which is discussed in Sec. 5, can match the machine to the model within a range within which the regulator can work well and have margin for drift.

7 Digital processor architecture

7.1 Computation for the time step

At this point we must contend with the amount and complexity of computation that must be completed each 0.88-µs time step (for the NSLS-II model). Referring again to Figure 3, the processing involved is the four gain blocks *A*, *B*, *C*, and *D* representing matrix multiplies and accumulates. The table shows the number of multiplies in each block in the NSLS-II LQG regulator archetype.

Table 11

Matrix	Sub blocks	Multiplies
A	2x2x5 1x1x4	24
B	14x3	42
C	2x14	28
Total		94

The number of multiplies suggests that individual multipliers in hardware need to handle multiple multiplies. This may appear problematic, but there are 33 clock cycles available within which to get the job done. Figure 12 shows the sequence of computations of Eq. 1.

Please be aware of the ambiguity of the *u* and *y* symbolism of the plant inputs and outputs vs. the regulator inputs and outputs. There is a similar ambiguity of plant *A*-, *B*-, *C*-, and *D*-matrix symbols and the identical symbols of the regulator state-space model. Furthermore, while there are noise inputs *w* to the plant, these inputs are not present at the regulator. Remember that in this section we are talking about the regulator exclusively and references to the *A*, *B*, *C*, and *D* matrices and inputs and outputs *u* and *y* refer to the regulator only.

Figure 22 and Figure 23 show how the matrix computation can be sequenced in a serialize architecture where each row of *B* and *D*, and column of *A* and *C* is computed in a single clock cycle. All the state variables of x_n appear serially each LQG-sample cycle. The matrix product Cx_{n-1} is also computed serially, column by column. The state variables are remembered from the previous cycle.

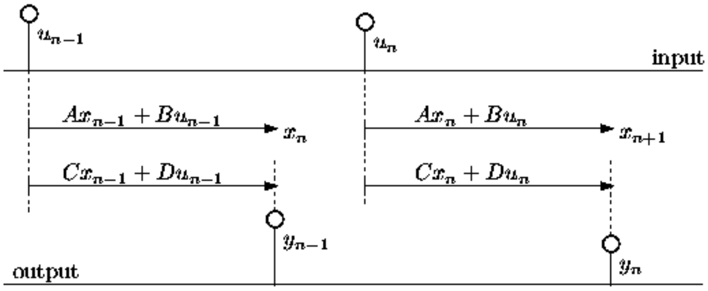

Figure 22: Sequencing of regulator inputs, outputs, and computations. In this diagram, no pipeline delays are assumed.

Figure 23 differs from Figure 22 in that the C matrix product proceeds with the newly computed x_{n+1} as they emerge from the *A* and *B* computation. This permits that matrix product to be computed one cycle sooner, and added to the Du_n product as soon as it is completed, which is generally sooner because the number of inputs is less than the number of state variables. Thus Figure 23 has lower latency than Figure 22.

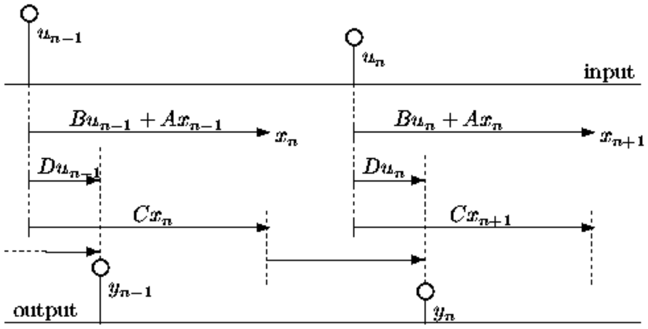

Figure 23: Sequencing of regulator inputs, outputs, and computations in an architecture with lower latency. This configuration takes advantage of the fact that the serialized *D* computation completes more quickly each cycle when there are fewer inputs than state variables. In this diagram, no pipeline delays are assumed.

Figure 24 shows a possible coarse architecture for the *A*, *B*, *C*, and *D* computations. In the *B* matrix block, the number of multipliers is the same as the number of columns of *B*. Three are shown. Coefficients are fed to the multipliers a row at a time while the inputs remain fixed. The outputs are fed to the sum (at the kernel summation point) at the same rate they are needed. So the *B* block is a simple matrix multiply of an ordinary dense matrix.

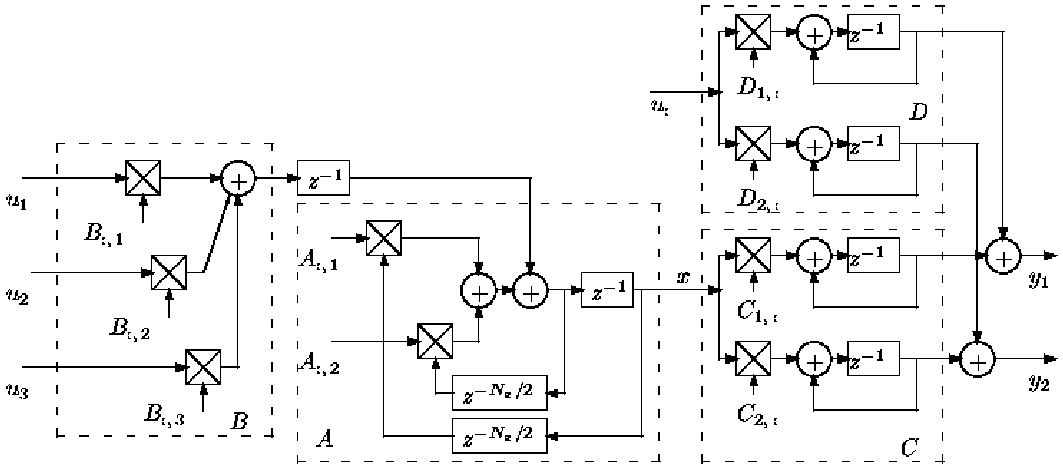

Figure 24: Regulator matrix computations.

The A-matrix block is different in that it has been arranged so that *A* is uniformly block diagonal with a block size of two. As was discussed earlier, this is because 2x2 blocks correspond to the space spanned by the eigenvectors of conjugate eigenvalue pairs, with the exception of real eigenvalues. For those real eigenvalues, LTICovR arranged that that they be paired, with the result that those 2x2 blocks are diagonal. (This works as long as the number of state variables is even.) The $z^{-N_x/2}$ are register stacks with depth $N_x/2$ clocked at half the data rate, where N_x is the number of state variables. Matrix coefficients are fed a two-element row at a time to the multipliers at the full data rate.

The C-matrix block is also an ordinary dense matrix multiply, this time in a multiply-accumulate configuration. Two outputs are shown in the Figure 24. Matrix coefficients are fed to the multipliers a column at a time, and the outputs valid at the end of the accumulate cycle. The z^{-1} registers are cleared at the beginning of the cycle. I also suggest two accumulator precision bits for this block, as was mentioned in Sec. 4.1.

The multipliers and register stacks have enable inputs for stopping computation when the regulator is idle, and when the computation is completed and it is waiting for new input samples. They also have reset inputs for use as described in Sec. 7.3**Error! Reference source not found.** on higher-level architecture.

Coefficients perhaps could be stored in small random-access memory blocks. An alternative is register stacks.

This particular configuration requires seven multipliers. These multipliers may be pipelined as needed and registers inserted as propagation delays warrant. Regarding processing delay from when an input becomes available to when outputs become available, a quick estimate is two pipeline delays plus the number of state variables (in clock cycles).

7.2 Prototype logic in Verilog

The architecture of Figure 24 was developed in Icarus Verilog [19] as a means to 1) test the concept and design of Figure 24 and 2) *begin* to flesh out the design. While the code verified Figure 24 through simulations and it can serve as a prototype upon which further development can be based, as it is it cannot run at the needed data rates. To do so it needs pipelining. As was mentioned earlier, it must also be meshed with the larger logic in a real controller that controls it and provides the matrix elements. Furthermore, a working system may be of a multi-cavity type discussed in Sec. 7.3. Discussed in this section is a single-cavity prototype only.

The code will be described quickly given that it is a prototype and not working code. The basic organization of elements of the logic is shown in Figure 25. The highest level element is `LQG.v`, the LQG regulator itself. In its mature form `LQG.v` might be instantiated as a unit in the larger cavity controller, i.e., in H. Ma's controller [7]. Other elements of the diagram are parameterized models instantiated by elements higher in the diagram. The files `ABlock.v`, `BBlock.v`, `CBlock.v`, and `DBlock.v` handle the computation of the matrix multiplies suggested by their names. Coefficient and data bit widths, overflows, and accumulator precision bits are as specified by parameters provided at build time. These parameters are exported by the function LTICovR in Matlab at the time the regulator is synthesized there.

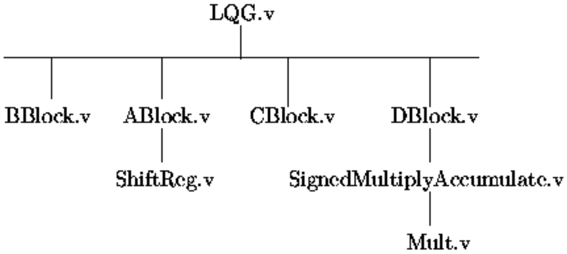

Figure 25: Verilog files and dependencies.

`Mult.v` provides a multiplier of specified multiplicand and output widths. There is rounding of the least-significant bit of the product. Rounding is important for reasons given in Sec. 4.1, although it is more relevant after a sum. `ShiftReg.v` provides a shift register of specified bit width and depth with synchronous shift, enable, and clear. `SignedMultiplyAcccumulate.v` functions as its name suggests with specified operand bit widths, output bit width, overflow bits, and accumulator precision bits.

The next three figures explicate the bit widths, locations of sign bits and binary points, etc., in the *A*, *B*, and *C* logic, beyond what is shown in Figure 5. Figure 26 is for the *A* matrix product, Figure 27 for the *B* matrix product, and Figure 28 for the *C* matrix product. Given the structure of the computation of Figure 24, for example, that *x* is both an input and an output of the *A*-matrix product, some of the parameters of the blocks are constrained by the other blocks. The diagrams hint at these constraints.

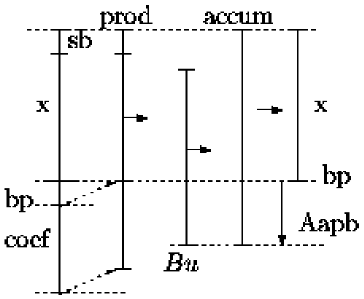

Figure 26: Coefficient and data bit widths, locations of sign bits (sb), binary points (bp), and accumulator precision bits (Aapb) for the *A*-matrix multiply/accumulate.

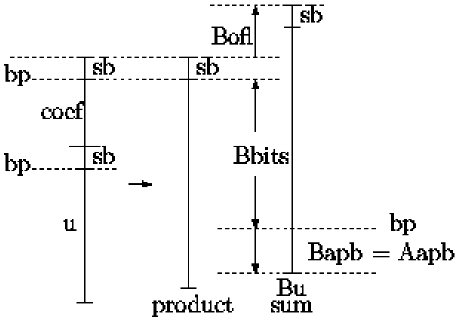

Figure 27: Coefficient and data bit widths, locations of sign bits (sb), binary points (bp), accumulator precision bits (Bapb), and overflow bits for the *B* matrix multiply/accumulate. Bbits is a *B*-matrix scaling parameter specified by `LTICovR`.

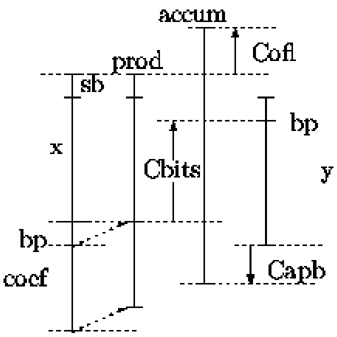

Figure 28: Coefficient and data bit widths, locations of sign bits (sb), binary points (bp), and accumulator precision bits (Capb) for the *C* matrix multiply/accumulate. Cbits is a *C*-matrix scaling parameter specified by `LTICovR`.

Synthesis of a regulator (in Matlab, not logic synthesis) would ordinarily take place when the controller measures the rf system response functions during a calibration time allocated for the purpose. This might be once a day. The response functions are then uploaded to a supervisory computer responsible for controller synthesis by the software in Matlab. At that time, parameters needed by `LQG.v` for logic synthesis of the specific LQG regulator are exported to a Verilog include file to be imported by `LQG.v`. From here, one of two things can happen.

- First, if bit counts within the LQG regulator change from synthesis to synthesis, then the controller logic must be resynthesized, and both regulator logic and coefficients must be downloaded to the rf controller at calibration time.
- Or if all the bit counts are stable from synthesis to synthesis, then the controller logic need not be resynthesized, and coefficients only need be regularly downloaded. The controller itself need be downloaded much less frequently (at a boot time).

Either way, the regulator synthesis software in Matlab and the prototype logic together outline how machine measurements by the controller, LQG regulator synthesis, and logic synthesis can be melded into a system that smoothly and periodically generates regulators and updates coefficients.

7.3 *Integration with a larger rf controller*

Before going into the more detailed look at the DSP architecture, first we look at how an LQG regulator might fit into a working machine. Such a regulator by its nature is tuned to control at a specific operating point of the machine and may not be tolerant of large swings in the operating point or other changes in the machine's behavior. In fact these changes are inevitably present. Making the controller function properly without beam, during injection, or during top off may not be possible without a time-dependent controller built on a detailed and accurate model of the dynamics of the machine. This problem is likely tractable, but unnecessary, given that the performance gain provided by the LQG regulator is needed only at full current and full energy. Here we benefit from the simplicity of top-off operation. We only need a time-independent regulator tuned for only one machine state.

But going this route means that the LQG regulator is not available at other times and an alternate one must be available. Most machines have a simple analog or digital proportional-integral (PI) regulator fixing rf cavity field intensity and phase. It may or may not be very effective at suppressing beam noise near and above the synchrotron frequency. For our purposes, it need only be stable at low and high field intensities and at all beam currents. There must also be a smooth crossover between the PI regulator and the LQG regulator.

So I suggest we consider the higher-level architecture of Figure 29.

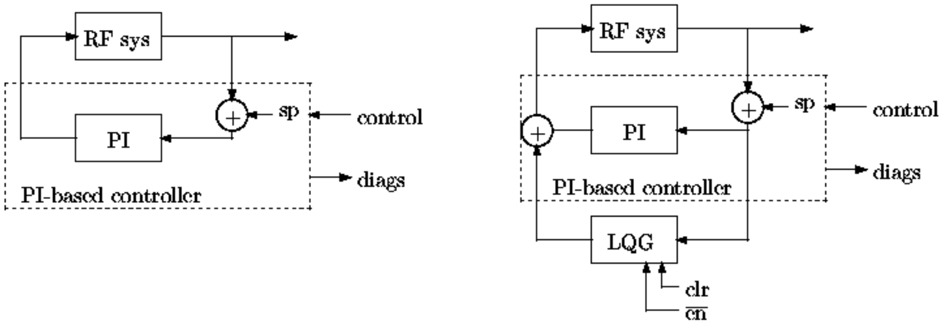

Figure 29: PI regulator (left) and LQG regulator in parallel with PI regulator (right). The PI block is a proportional-integral regulator and the LQG block is a linear-quadratic-Gaussian regulator.

Both the PI regulator and the LQG regulator have enable inputs that control whether they do anything. Only one is active at a time, as is indicated by the complementary states of the enable inputs indicated in Figure 29. So when one is active, the output of the other is frozen, the latter providing a fixed offset about which the former operates (Figure 30). When the cavity is idling, being ramped to power, during injection, or during top off, the PI regulator is active. The PI regulator has no use during this time for a control offset provided by the LQG regulator, so the state variables of the latter are reset at some time during this time. After injection and transients die away, the PI regulator is disenabled and the LQG regulator is enabled, taking control. The PI regulator is not reset and instead defines the nominal operating point for the LQG regulator for the duration. The LQG regulator need not provide a lot of effort to the plant, but instead only provide small noise-driven excursions about the operating point, excursions that are presumably effective at suppressing beam noise. Periodically during top off, control is switched to the PI regulator, the LQG regulator reset, and control switched back to the LQG regulator. The reset ensures that the PI regulator follows drifts in the operating point instead of the LQG regulator making up the difference. The system would be configured to automatically trigger switchover upon detection of a range of faults, such as a large excursion, a beam dump, etc.

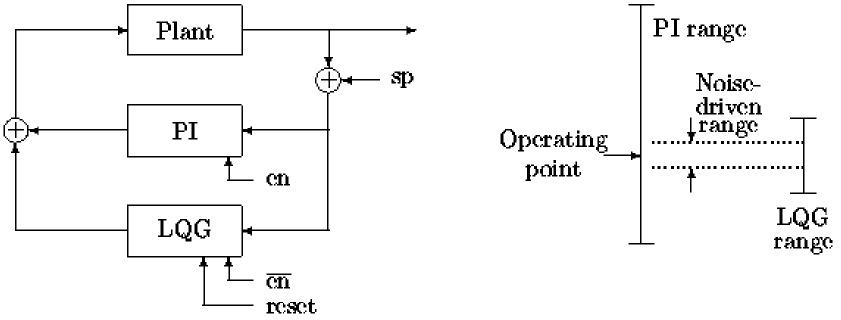

Figure 30: PI and LQG regulator ranges and operating point.

Using this scheme, the output range of the LQG regulator need not be as large as the PI-regulator's range. The range needed is poorly defined at this point since it depends on the magnitude of drift the machine realizes. But it may be quite small if top off occurs every couple of minutes since the noise-driven range is rather small. The utility of moderating the Kalman's output range is in the economy of the output signal processing (block C).

The PI/Kalman enable logic signal may be tied into the user gate signal to ensure that the LQG regulator is active when sensitive users are taking data.

The PI regulator would be on the same chip as the LQG regulator, but it need not consume significant chip resources due to its simplicity and low update rate.

This architecture may make possible testing of unstable LQG regulators by switching over to the LQG regulator for very short intervals. Intervals may be chosen that are sufficiently short that beam will not be lost even if the system is unstable.

Should there be drift in amplifier saturation resulting in excessive changes in amplifier (incremental) gain, this variation could be compensated for during operation with the `InAM` gain parameter placed at the output of the LQG regulator. Amplifier saturation could be periodically measured at the end of top off but before the LQG regulator is gated on. The measurement is done with a single-frequency response measurement using a subset of the logic of Figure 13. This is done by measuring the magnitude of the AM-to-AM matrix element of $T \cdot (1 - G)$ for the *S*-to-*a* (the forward wave output by the amplifier) response function. This measurement need only be done at one frequency chosen well below the synchrotron frequency. Since there is significant post processing, it must be done on a supervisory computer and would perhaps be done less frequently than every top off.

It should be noted that the `InAM` gain parameter compensates the regulator feedback path, but not the rf feedback path. For this reason it is not clear how well it compensates for amplifier saturation. An attenuator in that path linked to the `InAM` gain parameter may be needed. Depending on how deeply into saturation the amplifiers operate, saturation may complicate the operation of rf feedback due to the asymmetry in amplitude and phase it introduces [20]. It is true that direct rf feedback can be absorbed into the regulator, in which case the problem of compensating for amplifier saturation can be handle completely by the `InAM` gain parameter.

7.4 Distributed architecture for multiple cavities

Before considering the larger problem of scaling the regulator formalism developed in this report to the multi-cavity control problem with scaled degrees of freedom, it is appropriate to point out how to apply a single-cavity LQG regulator to a multi-cavity system. By this is meant simply taking feedback from the coherent sum of the cavities and applying the regulator's output to all the amplifiers. This does not control all degrees of freedom in the systems in that additional loops are needed to ensure that the systems keep the cavities' fields close to the same value. This is relatively simple to do with feedback using coherent detection of coherent differences of the cavity fields. If Σ/Δ couplers are used to form the coherent sum, then the coherent differences are available from these same couplers. These difference loops, if tuned properly, have minimal coupling to the beam. Configured this way, the rf systems appear to the LQG regulator as a single-cavity rf system.

We proceed now to the larger problem. With multiple cavities, the full state-space model has state variables for each amplifier and cavity. The cavity part of the model is thus duplicated, although they are distinct because they have different characteristics. As a consequence, the number of state variables is roughly multiplied by the number of cavities. The computational load, being matrix multiplies, scales, in most cases, with the square of this multiplicity, particularly the *B*, *C*, and *D* computations. But due to the block-diagonal structure of the kernel, the largest, the *A* portion, scales only linearly. It is the subject of this section to work out an architecture that distributes the computational load among the planned LQG regulators in the rf stations controlling the cavities and with a system-level controller in a natural and manageable way.

First note that because the actuators controlled by the LQG regulator reside in the rf stations (the I and Q modulators) all rows of the matrices *C* and *D* are specific to individual cavities. Therefore, computation of rows of these matrices is naturally assigned to their cavity regulator. There are no leftover rows. Similarly, rf cavity I and Q field and klystron forward and reverse signals I and Q components originate from the cavity and cavity controllers, so the columns of the matrix *B* that are associated with individual cavities are also naturally assigned to their cavity controllers. Unlike the *C* and *D* matrices, there are columns of the *B* matrix for beam signals (energy and/or phase). So the computation of each of these three matrix multiplies, *B*, *C*, and *D*, with the exception of columns of *B* associated with beam degrees of freedom, is divided uniformly among the cavity DSPs. The columns of the *B* matrix associated with beam signals, in contrast, do not naturally belong to a cavity controller. They are assigned to a system-level LQG regulator unit.

As was discussed earlier, the *A* block, through kernel transformations, is reduced to block-diagonal form with at most two-by-two blocks. This means that the *A* matrix multiply scales only linearly with the number of cavities plus the fixed number of beam degrees of freedom. Furthermore, due to the kernel transformations, there are no blocks of *A* that are associated with any one of the rf cavities. So one cannot identify a part of the kernel that naturally resides with any one rf cavity. Nevertheless, distributing the *A* computation among the cavity controllers can still be done while adding only modest additional computational load to those controllers and keeping the cavity controllers identical in form and function. First, blocks of the *A*-matrix multiply corresponding *in number* to the beam degrees of freedom are assigned to the system controller. Second, the rest of the *A* blocks are divided uniformly among the cavity LQG regulators. In this way, the *A*-computational load is naturally distributed uniformly among the cavity controllers, which have identical logic implemented in them (but different coefficients), and the system controller, which has only a modest computational workload in addition to its signal-distribution function.

While the computational intensity of the *A* block in each regulator is not greater than in a single-cavity LQG regulator (less actually), remember that the number of columns of *C* and *D* and the number of rows of *B* have been scaled, aside from the beam degrees of freedom, by the number of cavities. So the workload of the *B*, *C*, and *D* blocks in each of the cavity regulators still scales with the number of cavities. So the *B*-, *C*-, and *D*-matrix computations of the cavity regulators require parallelization by the number of cavities beyond that of Figure 24.

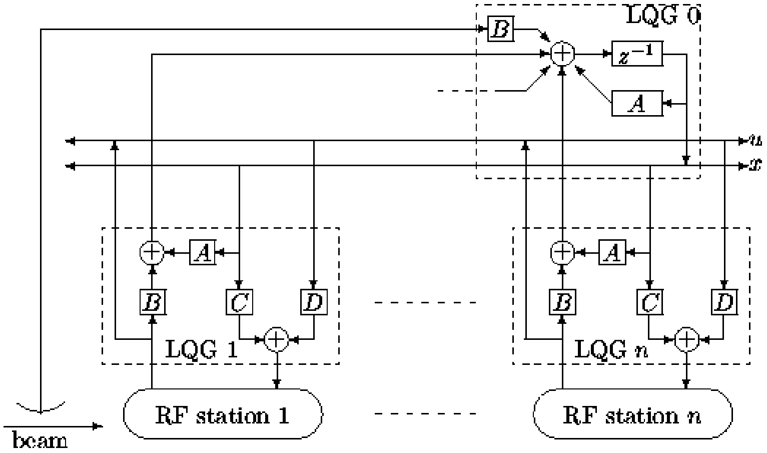

Figure 31: Architecture for a multi-cavity distributed LQG regulator computation.

Figure 31 shows the architecture just outlined. Remember that the *A* and *B* blocks of the system LQG 0 regulator differ in form from those in the cavity regulators, while the *A*, *B*, *C*, and *D* blocks in the cavity regulators are identical in form, but differ in coefficients.

The timing sequence is roughly shown in Figure 32 and builds on the single-cavity cycle of Figure 22. At the time indicated by u_{n-1}, the inputs become available from the ADCs in the cavity controllers. The state variables of the previous cycle had been serially transmitted so that the first components of x_{n-1} arrive at the cavity LQG regulators at the same time the inputs u_{n-1} became available there. The *A* and *B* computations then proceed serially and the results are transmitted serially to LQG 0 as they are completed. As those results arrive, the new x_n are serially computed, latched at LQG 0, and serially transmitted to the cavity LQG regulators for the *C* and *D* computations. Those computations proceed serially as the components of x_n arrive. At the end of the cycle, the new outputs y_n are available at the cavity controllers. Thus the signal transit times are integrated into the computation cycle in a simple way with minimal delay.

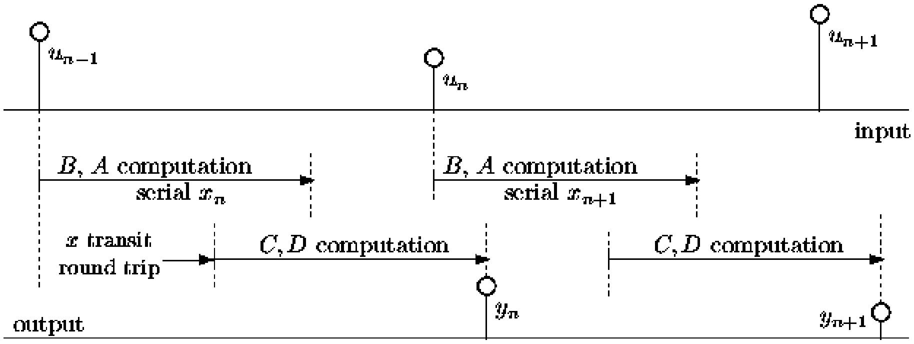

Figure 32: Sequencing of computations occurring in the multi-cavity architecture of Figure 31.

A very rough estimate of the round trip delay is 300 ns, given the relative locations of the two rf straights. Data transmission is configured as a stream that is initiated once per cycle in each direction. It must not be buffered, or else a great deal of latency is added. The data transmitted must be padded with zeros between data words, because the *x* data rate is less than the capacity of the type of link likely to be used. Configured as a stream, the minimum delay introduced by the data link is the physical transit time; even the setup times can be anticipated.

The *u* data links that feed the *D* computations in the cavity LQG regulators function in principle similarly to the *x* data links. But only a smaller quantity of data must be transmitted to the cavity regulators in time for the start of the *D* computation. Because the *D* computation is shorter and need only be finished by the time the *C* computation finishes, the *u* data links do not add additional delay to the *y* computation.

Pipeline delays are not shown in Figure 32. They add delay between the *B*/*A* computations and the *C* computation, and extend the lengths of the B and A computations, and the C computations. Depending on the details of the serial-link interfaces, minimal buffering at the inputs and outputs of the serial links may be required.

Although the regulator logic is not fleshed out beyond this point, there are a number of points to be made to make these ideas more understandable. Sixteen-bit samples, four rf stations, *u* signals `VI` and `VQ` from each rf station, a single beam sample (*tau* presumably), and an LQG data rate of 2 MHz are assumed for the serial data-rate estimates below.

- The kernel summation point, sub-blocks of the *A* computation corresponding in multiplicity to the beam degrees of freedom, and columns of *B* (the *B* block in LQG 0) corresponding to the beam (energy and/or phase) inputs reside on the system LQG-0 regulator as shown. This for logical consistency and to ensure that the cavity LQG regulators are identical in form and function.
- The single-cavity regulator whose construction is described in Secs. 3 and 4 is intended to run at the rf controller's clock rate of ~ 40 MHz, with a data rate of 1-2 MHz. With that clock rate there are some spare clock cycles at the end of the LQG cycle. The multi-cavity LQG regulators' clock rate is assumed at the same ~40 MHz and that that computation must be paralleled, with the multiplicity being the number of cavities, and with a similar number of spare clock cycles at the end of the LQG cycle. Clock cycles for any additional pipelining will eat into that spare.
- The multiplicity of cavities has an impact on the number of extra accumulator bits and stages of pipelining needed in the various computations through the number of terms in the matrix multiplies, as described in Sec. 4.1.
- Delay due to the physical separation between controllers on different rf straights has an impact on the rf model from which the regulators are synthesized. It also has an impact on the timing of the logic, as does the transmission delays on the *u* and *x* data links. Additional delays affect the model if additional sample times are added to the regulator inputs of the rf model (Sec. 3.2).
- The single-cavity regulator requires seven multipliers of a few hundred logic elements each to implement. With four cavities and, say, 500 logic elements per multiplier, each cavity LQG regulator needs something like ((3+2)*4+2)*500 = 12,000 logic elements for multipliers. Further parallelization increases this number.
- All of the LQG regulators have, or work with, embedded network analyzers in logic in the manner described in Sec. 5.2. Measurement of off-diagonal blocks of the system response function using these network analyzers

requires synchronized data acquisition among the controllers. Fitting the multi-cavity version of the model of Figure 2 to these response data (Sec. 5) might present a significant computational challenge.

- Distribution of signals by the system controller is presumably by high-speed serial links, perhaps through fiber optics, although twisted pair may be adequate within a straight section.
- The *u* (inputs) bus must be combined from signals from the beam and cavity controllers and is distributed to the cavity controllers. From each cavity LQG regulator, the *u* data rate is 2 real samples per 0.5-1 μs sample interval ~ 60 Mbps average rate.
- All *u* (input) samples are needed for the *D*-matrix computation, which scales with the number of cavities. Thus a data rate of nine samples per LQG sample interval ~ 0.3 Gbps average data rate is needed to carry the *u* samples to each cavity LQG regulator.
- The *x* (state-variable) signals are distributed to the cavity LQG regulators for the *C*-matrix computation and demand the greatest data-carrying capacity of the serial links. The data rate is ~50 state variables per LQG sample time ~ 1.5 Gbps average rate on each link.
- The *A*+*B* link from each cavity controller to the system controller carries roughly 10 real samples each LQG sample time, or ~0.3 Gbps average data rate.
- These data-rate and computation estimates have assumed that cavity I and Q signals (`VI` and `VQ`) only from each cavity are used for feedback. The use of additional signals, such as the forward (aI and aQ) and reflected (bI and bQ) waves, for feedback

8 Notes and conclusions

Models of single-cavity rigid-bunch rf systems were developed to address the tight beam-noise requirements of timing and FTIR experiments at NSLS-II. Numerical simulations of LQG regulators in these models were found to realize a great deal of gain and bandwidth and show a corresponding degree of noise suppression of amplifier noise compared to simpler proportional-integral regulators. If this performance can be realized in an rf system, then that machine can tolerate considerably more rf-system noise than otherwise.

Although these regulators are fine tuned to the characteristics of a particular model, they do not show a prohibitive amount of sensitivity to variation of the model. This is encouraging and to a degree allays fears about the difficulty of fabricating robust regulators. But machine measurements are a necessity for verifying by simulations that other degrees of freedom of the machine, such as higher-order bunch multipoles, can be neglected. To a degree, those concerns are allayed by the success of an LQG regulator synthesized for a Figure 2-model fitted to Vlasov-calculated response functions, controlling a state-space model faithfully fitted to the same Vlasov response functions. Machine measurements are also needed to fine tune the regulator for optimum performance and margin of error.

RF-system and regulator models were developed for NSLS-II, CLS, and NSLS's VUV rings. Models of all of these machines yielded apparently viable regulators and similar performance improvements. The VUV and CLS models show better natural (open-loop) suppression of amplifier noise than does the NSLS-II model.

It was discussed in Sec. 3.5 that the synthesis of floating-point regulators in Sec. 3 often results in unstable regulators, which often have higher gains and higher closed-loop performance. It was suggested that stable regulators be chosen because of complications using unstable ones. But with digital logic, it is feasible to test for closed-loop stability by switching the regulator on and then off after a short time, and Fourier analyzing signals in the loop. This is performed in the manner of coupled-bunch mode growth-rate measurements in bunch-by-bunch feedback systems [21], where growth of an instability is detected and cut off before significant amplitudes are reached. Where robust higher-gain regulators can be tested this way successfully, improved performance may result. Alternatively, the choice of moderate-gain unstable regulators using, for example, transmission line signals `aI`, `aQ`, `bI`, and/or `bQ,` may not result in improved performance, but may provide more robust regulators by other measures. This is a direction that can be explored with digital controllers.

Methods for establishing resolutions of the coefficients and data paths of fixed-point regulators were developed. It turns out that these regulators require relatively modest resolutions, easing the resource requirements of a digital implementation.

These results are very encouraging in that they provide evidence that viable regulators for rf systems in top-off operation can be constructed and that their performance is exceptional in suppressing amplifier noise. A number of tools for construction and testing of these regulators are also demonstrated. Fixed-point regulators built from these tools seem to work well and are realizable in modest FPGAs. Further work is needed to better characterize amplifier noise, measure machine characteristics including response functions, and to fill in many details of how these regulators would be meshed with the larger rf control system, particularly with multiple cavities.

9 Acknowledgements

Thanks to Jim Rose for many helpful discussions, and Hengjie Ma for his digital controller and the exceptional data it records.

---

[1] Brookhaven National Laboratory, *National Synchrotron Light Source II Preliminary Design Report*, 'http://www.bnl.gov/nsls2/project/PDR/' (November 2007).

[2] H. Hindi et al., *A Formal Approach to the Design of Multibunch Feedback Systems: LQG Controllers*, Presented at the Fourth European Particle Accelerator Conference, London England (June 1994).

[3] Paul Moroney, *Issues in the Digital Implementation of Control Compensators*, Massachusetts Institude of Technology dissertation (October 1979).

[4] D. Boussard and E. Onillon, Application of the Methods of Optimum Control Theory to the RF System of a Circular Accelerator, CERN Report No. CERN SL/93-09 (RFS) (2 Mar 1993).

[5] N. Towne, Measurement of the Generator Impedance Through the Longitudinal Beam Response Function, unpublished NSLS report (November 2001).

[6] T. Mastorides, C. Rivetta, J. D. Fox, and D. Van Winkle, Analysis of longitudinal beam dynamics behavior and rf system operative limits at high-beam currents in storage rings, Phys. Rev. ST-AB **11**, 062902 (2008).

[7] H. Ma and J. Rose, Low-Level Radio Frequency System Development for the National Synchrotron Light Source II Project, 23rd Particle Accelerator Conference, Vancouver, BC (2009).

[8] IEEE, *IEEE Standard for Verilog Hardware Description Language*, IEEE Std 1364-2005, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1620780.

[9] The Mathworks, Inc., Matlab version R2009a (2009).

[10] The MathWorks, Inc., © 1994-2008, http://www.mathworks.com.

[11] H. Hindi, Background Notes for Control Theory with Application to Accelerators, from lectures given at UC Berkeley (January 1997).

[12] H. Padamsee, J. Knobloch, and T. Hayes, *RF Superconductivity for Accelerators*, Wiley, NY (1998), sec. 17.5.2.

[13] H. Ma, private communication (2010).

[14] N. Towne and J.-M. Wang, Phys. Rev. E **57** (3), p 3461 (1998).

[15] N. Towne, Phys. Rev. ST-AB Vol. **4**, 114401 (2001).

[16] N. Towne, Longitudinal Bunch Profiles, RF Response Functions, and Coupled-Bunch Instability Thresholds in NSLS-II with Normal- and Super-Conducting RF Cavities, for NSLS-II under BSA contract number 117376 (August 2007).

[17] Canadian Light Source Inc., 101 Perimeter Road, Saskatoon, SK, Canada S7N 0X4, http://www.lightsource.ca.

[18] D. Van Winkle et al., *Klystron Linearizer*, presentation before the PEP-II MAC Review of 19 Jan 2006, Stanford Linear Accelerator Center (2006).

[19] Stephen Williams, Icarus Verilog, http://www.icarus.com/eda/verilog.

[20] J. Fox et al., *Lessons learned from PEP-II LLRF and Longitudinal Feedback*, presentation WEOBM02 at 11th European Particle Accelerator Conference, Genoa, Italy (June 2008).

[21] D. Teytelman et al., *Control of Multibunch Longitudinal Instabilities and Beam Diagnostics Using a DSP-Based Feedback*, 1997 Particle Accelerator Conference, Vancouver, B.C., Canada, p. 2284-6 (12-16 May 1997).