

Formal Methods for Control Engineering
Progress Report

March 2002

Richard Boulton, Ruth Hardy, Tom Kelsey, Ursula Martin

School of Computer Science
University of St Andrews

North Haugh
St Andrews
Fife KY16 9SS
Scotland, UK

Outline

- Background, Objectives and Approach
- Examples
- Closed-Loop Transfer Functions
- Stability Criteria
- Preliminary Results and Conclusions
- Formalisation and Automation
- Possible Directions for Further Work
- Questions and Topics for Discussion

Machine-Assisted Reasoning for Computational Mathematics

- Machine-assisted reasoning produces formal proofs in a rigorous logical system
 - e.g. HOL (Cambridge), PVS (SRI), ProofPower (Lemma 1)
 - applied by Intel, Motorola, QinetiQ to critical applications
- Our goal:
 - machine-assisted reasoning to enhance computational mathematics
 - use familiar informal systems (Maple, MATLAB, NAG) as normal
 - generate proof obligations for theorem prover to increase
 - * assurance
 - * scope

Some Approaches

Maple-PVS

- use computer algebra system to compute and experiment
- pass conjectured result to PVS theorem prover

proof obligation: computed result

Larch-AXIOM (sponsored by NAG Ltd)

- “smart comments”: annotate legacy numerics components with assertions
- use in verification, program synthesis, method selection

proof obligation: property of computed result

Verified table look-up

- theorem prover decides which case of symbolic table holds

proof obligation: satisfiability of symbolic constraints

Formal Methods for Control Engineering

Work at QinetiQ

- Formal verification of critical systems
 - “formal”: rigorous, detailed, mechanisable
- ClawZ: Simulink models \rightarrow Z specifications
- Compliance Tool for annotated Ada code
 - verification conditions: establish using a proof assistant
- Want additional acceptance criteria

Variety of other approaches

- DARPA Mobies project (CMU, Berkeley, SRI, Ford, ...)
- Hybrid systems, model as state machine

Objectives and Approach

Objective:

- Investigate how machine-assisted reasoning might be applied in control engineering

Approach:

- Learn some control engineering
- Determine possible proof obligations
 - e.g. relationship between stability in continuous-time and discrete-time models
 - initially for tractable textbook examples
- Study feasibility of machine-assisted reasoning
 - focus on *discrete-time* systems

Examples

1. Simple cruise controller using a PID controller:

$$G_p(s) = \frac{1}{ms + b} \quad G_c(s) = K_p + \frac{K_i}{s} + K_d s$$

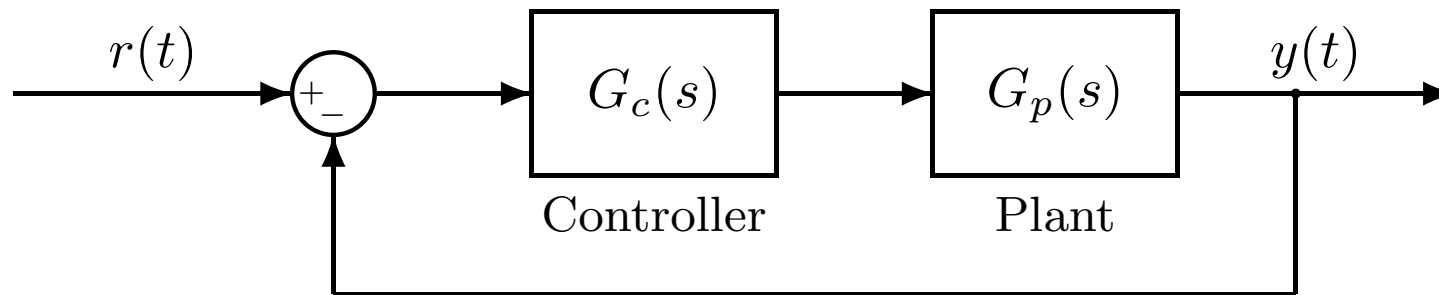
where m is the mass, and b is the coefficient of friction

2. Fairly generic second-order system controlled by a PD controller:

$$G_p(s) = \frac{K}{s(\tau s + 1)} \quad G_c(s) = K_p + K_d s$$

where K and τ are constants/parameters

Continuous-Time Closed-Loop Transfer Function

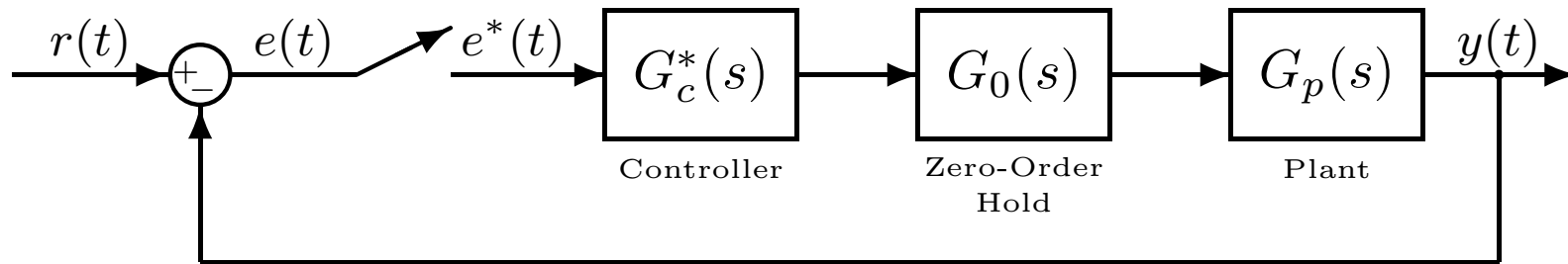


$$\frac{Y(s)}{R(s)} = \frac{G_c(s)G_p(s)}{1 + G_c(s)G_p(s)} = \frac{\dots}{q_{\text{cont}}(s)}$$

E.g.

$$\frac{Y(s)}{R(s)} = \frac{K_d s^2 + K_p s + K_i}{(K_d + m)s^2 + (K_p + b)s + K_i}$$

Discrete-Time Closed-Loop Transfer Function



$$G_0(s) = \frac{1 - e^{-sT}}{s}$$

$$G_{0p}(s) = G_0(s)G_p(s)$$

$$\frac{Y(z)}{R(z)} = \frac{G_c(z)G_{0p}(z)}{1 + G_c(z)G_{0p}(z)} = \frac{\dots}{q_{\text{disc}}(z)}$$

Stability

- Characteristic polynomial q : denominator of transfer function
- Conditions for absolute stability:
 - Continuous: all roots of q_{cont} must lie in left half of complex plane
 - Discrete: all roots of q_{disc} must lie within the unit circle
- Do not need to find the roots of the polynomials
 - can determine from coefficients whether conditions hold
- The above is for absolute stability
 - conditions need to be refined for other aspects of stability
- We want to relate the continuous and discrete conditions

Routh-Hurwitz Stability Criteria

- General method for determining continuous stability conditions from characteristic polynomial of degree n
- Stability conditions for a cubic polynomial $as^3 + bs^2 + cs + d$:
($a > 0$ **and** $b > 0$ **and** $c > 0$ **and** $d > 0$ **and** $bc > ad$) **or**
($a < 0$ **and** $b < 0$ **and** $c < 0$ **and** $d < 0$ **and** $bc > ad$)
- Conditions are necessary and sufficient for absolute stability (for all sizes of polynomial)

Jury Stability Criteria

- General method for determining discrete stability conditions from characteristic polynomial
- Stability conditions for a cubic polynomial $az^3 + bz^2 + cz + d$ when a is positive:
 $|d| < a$ **and** $a + b + c + d > 0$ **and** $-a + b - c + d < 0$ **and**
 $|d^2 - a^2| > |bd - ac|$
- Analogous conditions exist for negative a
 - or can normalise polynomial to have 1 as its leading coefficient

Stability of Example 1 (Cruise Control)

- Characteristic polynomial for continuous model:

$$(K_d + m)s^2 + (K_p + b)s + K_i$$

- Making leading coefficient 1, Routh-Hurwitz criteria are:

$$\frac{K_p + b}{K_d + m} > 0 \quad \text{and} \quad \frac{K_i}{K_d + m} > 0$$

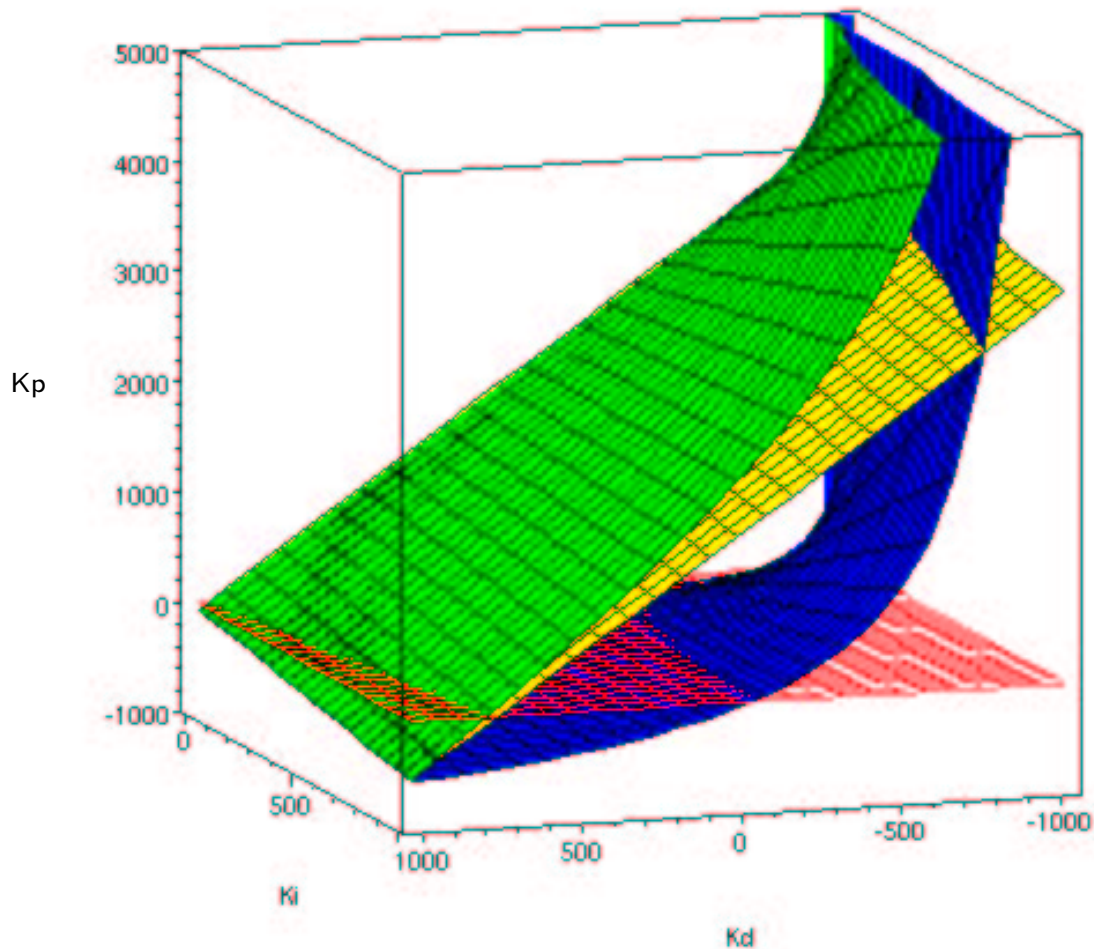
- Characteristic polynomial for discrete model:

$$bTz^3 + \left(\begin{array}{l} K_pT(1 - e^{-\frac{bT}{m}}) + K_d(1 - e^{-\frac{bT}{m}}) + \\ K_iT^2(1 - e^{-\frac{bT}{m}}) - bT(1 + e^{-\frac{bT}{m}}) \end{array} \right) z^2 + \\ (2K_d(e^{-\frac{bT}{m}} - 1) + K_pT(e^{-\frac{bT}{m}} - 1) + bTe^{-\frac{bT}{m}})z + K_d(1 - e^{-\frac{bT}{m}})$$

- Jury criteria too complex to show, so represent graphically ...

Discrete Stability (Jury) Criteria

$$T = 1, m = 1000, b = 50$$



Relationship:

- Region of discrete stability much smaller than region of continuous stability
 - but not a subset

Stability of Example 2 (Second-Order System)

- Characteristic polynomial for continuous model:

$$\tau s^2 + (1 + KK_d)s + KK_p$$

- Making leading coefficient 1, Routh-Hurwitz criteria are:

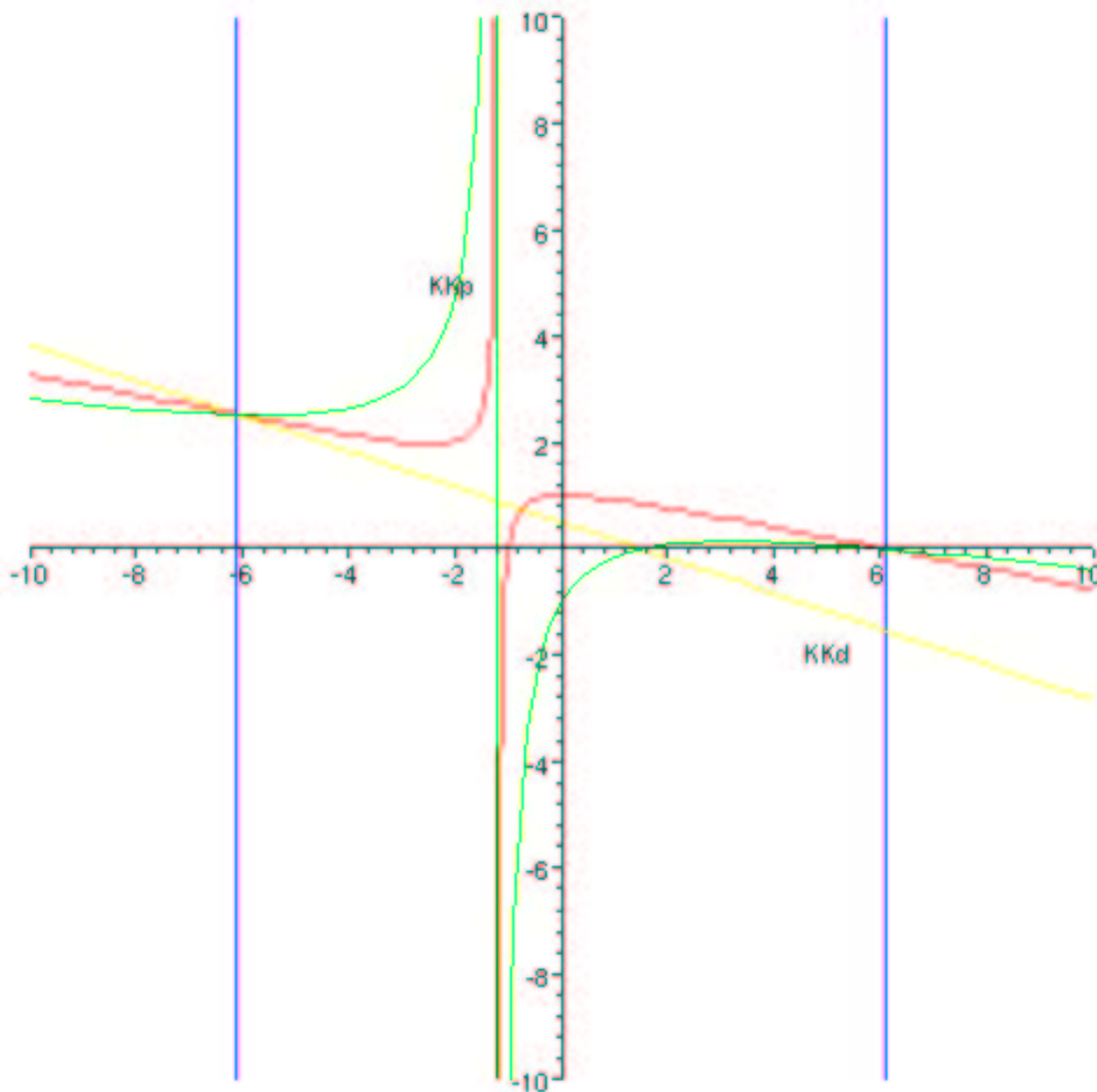
$$\frac{1 + KK_d}{\tau} > 0 \quad \text{and} \quad \frac{KK_p}{\tau} > 0$$

- Characteristic polynomial for discrete model:

$$z^3 + \left(K(K_p T + K_d) \left(1 - \frac{\tau}{T} (1 - e^{-\frac{T}{\tau}}) \right) - (1 + e^{-\frac{T}{\tau}}) \right) z^2 + \left(e^{-\frac{T}{\tau}} - K(K_p T + K_d) \left(e^{-\frac{T}{\tau}} - \frac{\tau}{T} (1 - e^{-\frac{T}{\tau}}) \right) \right) z + KK_d \left(e^{-\frac{T}{\tau}} - \frac{\tau}{T} (1 - e^{-\frac{T}{\tau}}) \right)$$

Discrete Stability (Jury) Criteria

$$T = 6, \tau = 1$$



- Discrete stability region is contained within continuous region, and is much smaller
- As sampling period T gets large, region of stability becomes very small

Preliminary Results and Conclusions

- Analysed stability criteria with the assistance of a computer algebra system (Maple)
- For our examples, range of values that are within stable region is much larger in continuous domain than in discrete domain
- For second-order example, discrete stability appears to imply continuous stability (when $\tau = 1$)
- For cruise controller, there are values that give discrete stability but not continuous stability
 - but the values involve a negative controller parameter
- So far, analysis has not been checked by mechanised formal proof

Possible Proof Obligations

Example general form (from Example 2):

For all values of $K, K_p, K_d, T,$

$$\text{Stability}_{\text{disc}}(K, K_p, K_d, T) \implies \text{Stability}_{\text{cont}}(K, K_p, K_d, T)$$

From Example 1:

For all values of $K_p, K_i, K_d,$

$$(K_p > 0 \text{ and } K_d > 0) \implies$$

$$\text{Stability}_{\text{disc}}(K_p, K_i, K_d) \implies \text{Stability}_{\text{cont}}(K_p, K_i, K_d)$$

Formalisation and Automation

- Proofs about stability criteria for specific systems:
probably tractable
- Formal verification of stability criteria themselves: major project
- Automatic derivation of transfer functions and stability criteria
 - prototype procedures have been implemented in Maple
- Integration into ClawZ

The Broader Landscape

- Other example systems
- Variants of controller
 - e.g. different ways to compute discrete integrals and derivatives
- Different ways of obtaining z-transform
 - e.g. *bilinear transformation*
- Different configurations of closed-loop system
- Other ways of analysing stability
- Other properties, e.g. rise time, overshoot, settling time
- Accuracy of model, e.g. propagation delays, bits of precision

Questions

- Are the continuous and discrete-time models developed by different people?
- If so, what is communicated between them?
- What determines the parameter settings used in the actual implementation?
- What would be the consequences of using parameters that give “discrete stability” but not “continuous stability”?
- Are there any theorems about the relationship between stability in the two time domains?
- What other properties are important?
- Is it sensible to allow negative PID controller parameters?

Discussion Topics

- What techniques are used in “real” design and analysis?
- Global vs local analysis
- Design vs post hoc verification
- “Documentation” for results of analysis to aid verification
- Adding assertions to Simulink and/or ClawZ