# Virtual Acoustic Musical Instruments: Review of Models and Selected Research

Julius O. Smith III (`jos@ccrma.stanford.edu`)
Center for Computer Research in Music and Acoustics (CCRMA)
Department of Music, Stanford University
Stanford, California 94305
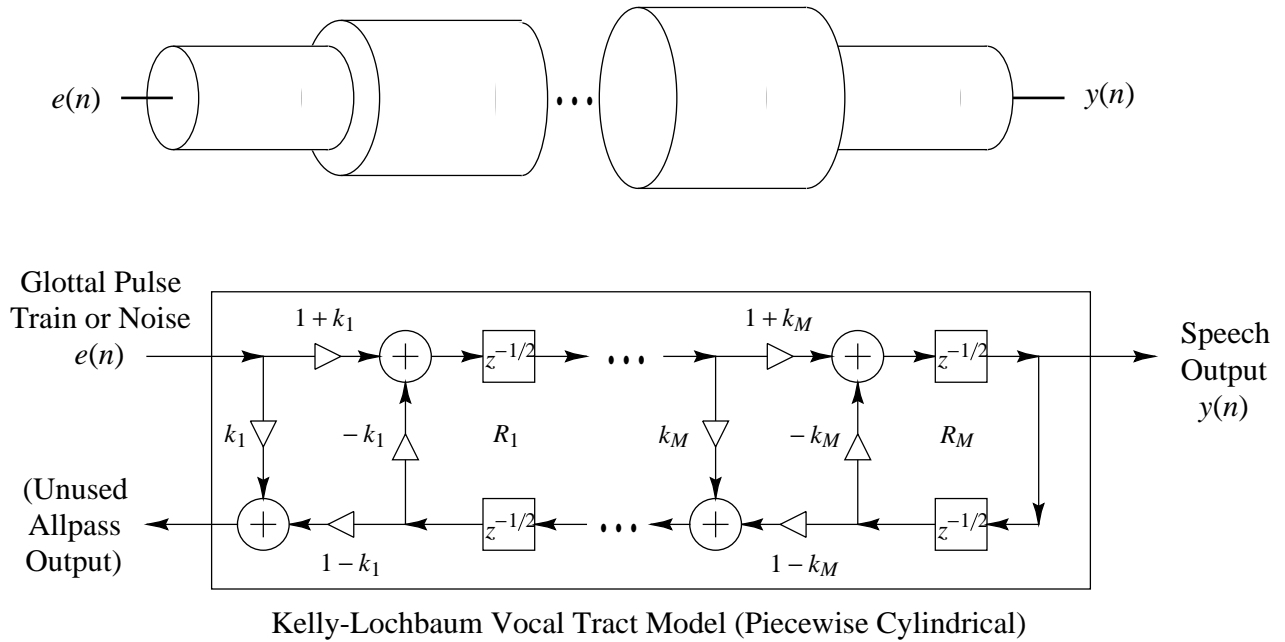
Oct. 19, 2005

Presentation Overheads

# Outline

In *historical order*, with selected updates:

- Voice Synthesis

- Karplus-Strong Algorithm

- Waveguide Synthesis

- Commuted Synthesis

- Virtual Analog

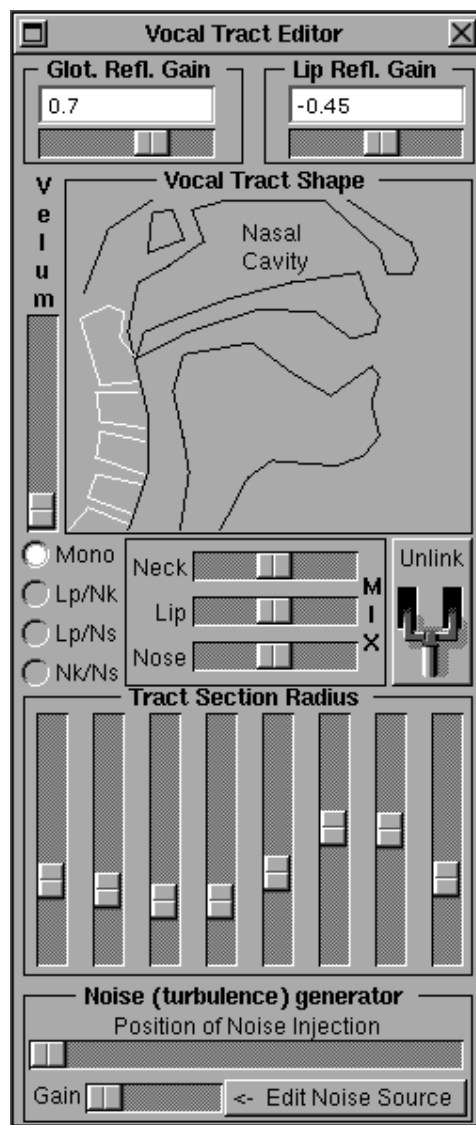# Voice Modeling

## Linear Prediction (LP) Vocal Tract Model



Kelly-Lochbaum Vocal Tract Model (Piecewise Cylindrical)

## John L. Kelly and Carol Lochbaum (1962)

# Sound Example

"Bicycle Built for Two": (WAV) (MP3)

- Vocal part by Kelly and Lochbaum (1961)

- Musical accompaniment by Max Mathews

- Computed on an IBM 704

- Based on Russian speech-vowel data from Gunnar Fant's recent book

- Probably the first digital physical-modeling synthesis sound example by any method

- Inspired Arthur C. Clarke to adapt it for "2001: A Space Odyssey" — the computer's "first song"

# "Shiela" Sound Examples by Perry Cook (1990)



- Diphones: (WAV) (MP3)
- Nasals: (WAV) (MP3)
- Scales: (WAV) (MP3)
- "Shiela": (WAV) (MP3)

# Recent Voice Modeling Efforts

## Linear Prediction (LP) Vocal Tract Model

- Can be interpreted as a modified Kelly-Lochbaum model

- In linear prediction, the glottal excitation must be an

  - *impulse*, or
  - *white noise*

  *This prevents LP from finding a physical vocal-tract model*

- A *more realistic glottal waveform* $e(n)$ is needed before the vocal tract filter can have the "right shape"

- How to augment LPC in this direction without going to a full-blown *articulatory synthesis model*?

  - Jointly estimate *glottal waveform* $e(n)$ so vocal-tract filter can have the "right shape"

# Klatt Derivative Glottal Wave

Two periods of the basic voicing waveform
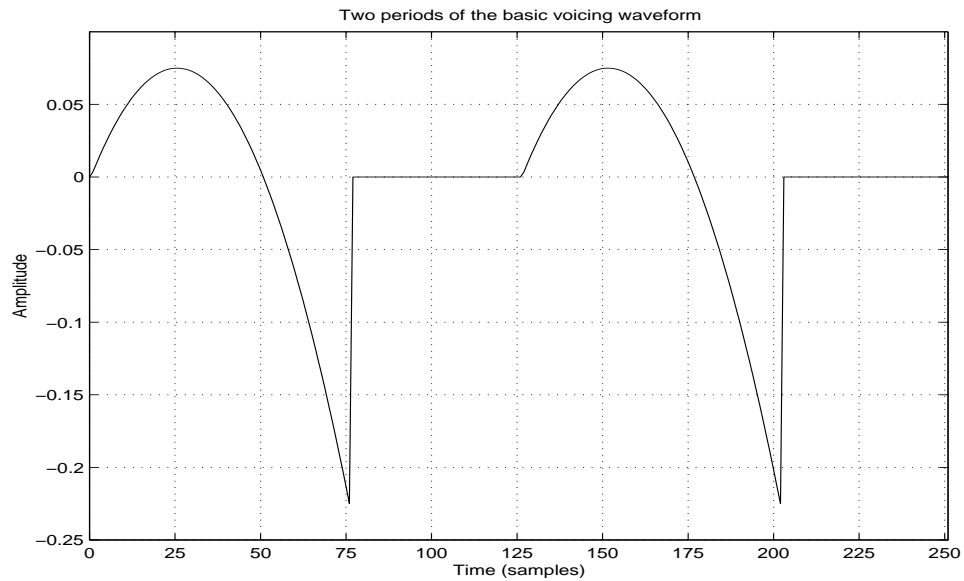


- Good for estimation:

  – Truncated *parabola* each period

  – Coefficients easily fit to *phase-aligned* inverse-filter output

# Sequential Unconstrained Minimization

## (Hui Ling Lu, 2002)

Klatt glottal (parabola) parameters are estimated *jointly* with vocal tract filter coefficients

- Formulation resembles that of the *equation error method for system identification*

- For *phase alignement*, we estimate

    - pitch (time varying)
    - glottal closure instant each period

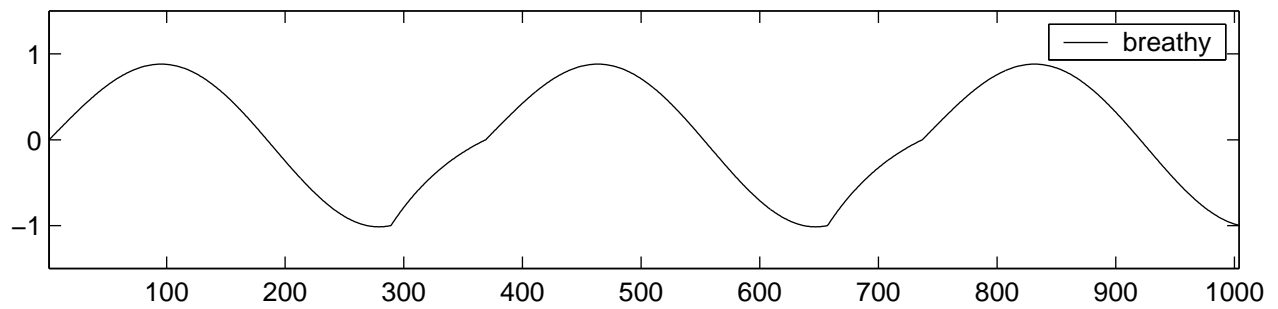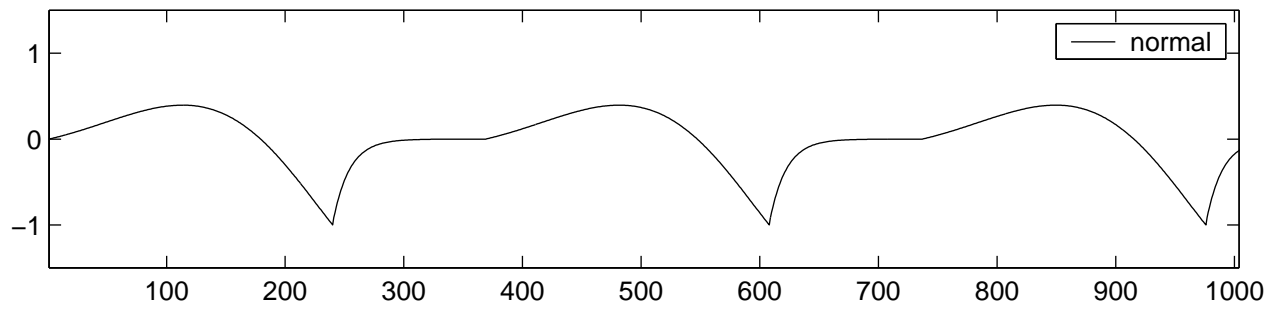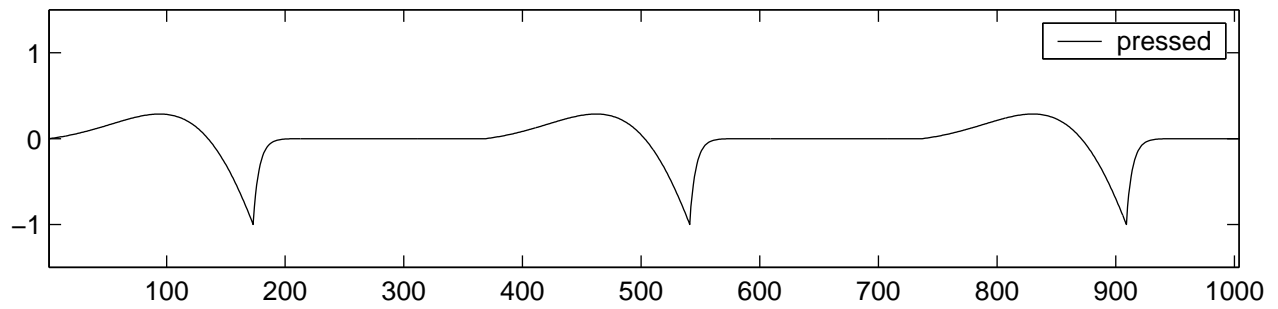- Optimization is *convex* in all but the phase-alignment dimension

# Liljencrantz-Fant Derivative Glottal Wave Model
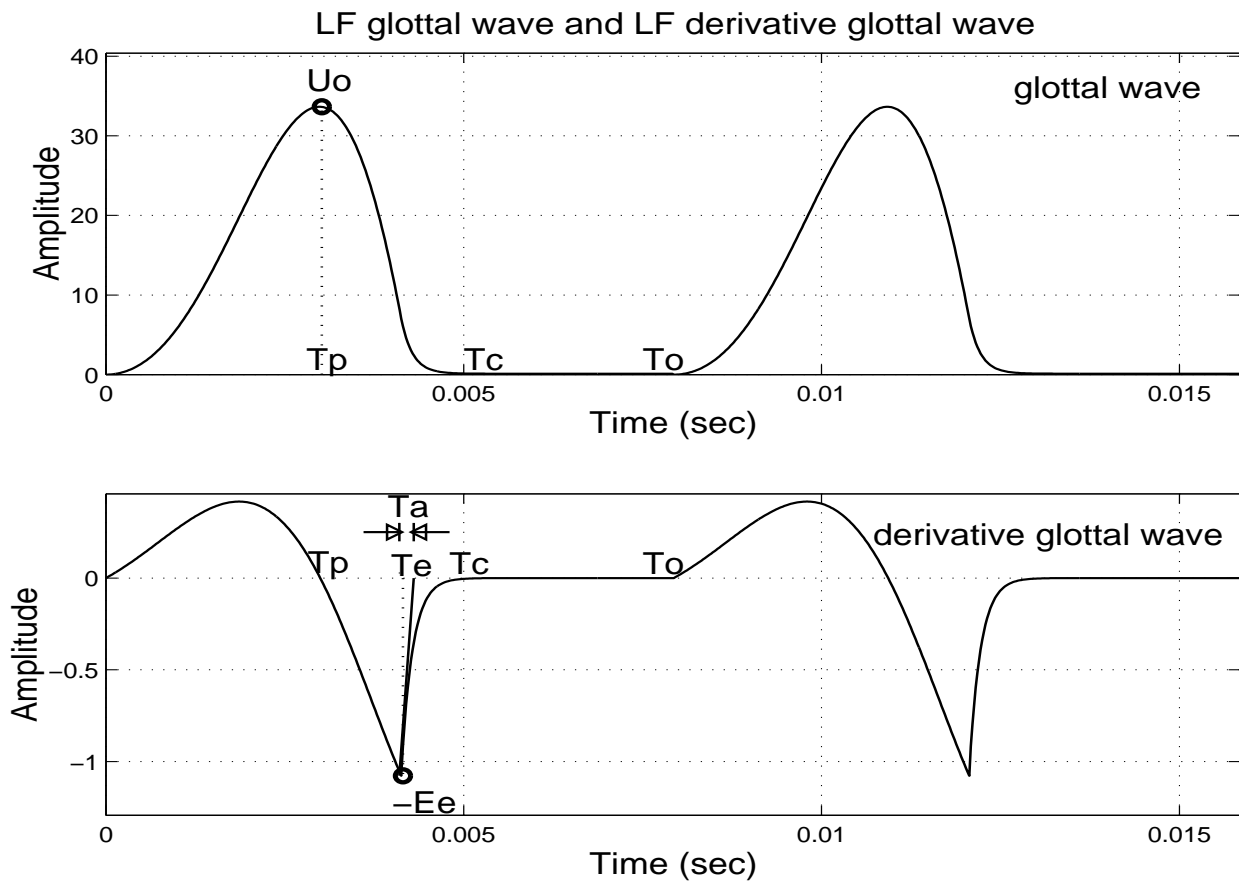


- Better for *intuitively parametrized expressive synthesis*

- LF model parameters are fit to *inverse filter* output

- Use of Klatt model in forming filter estimate yields a "more physical" filter than LP

# Parametrized Phonation Types

# Sound Examples by Hui Ling Lu

- Original: (WAV) (MP3)

- Synthesized:

    - Pressed Phonation: (WAV) (MP3)
    - Normal Phonation: (WAV) (MP3)
    - Breathy Phonation: (WAV) (MP3)

- Original: (WAV) (MP3)

- Synthesis 1: (WAV) (MP3)

- Synthesis 2: (WAV) (MP3)

where

- Synthesis 1 = Estimated Vocal Tract driven by estimated KLGLOT88 Derivative Glottal Wave (Pressed)

- Synthesis 2 = Estimated Vocal Tract driven by the fitted LF Derivative Glottal Wave (Pressed)

Google search: *singing synthesizer vocal texture control* (Hui Ling Lu's thesis page at CCRMA)

# Voice Model Estimation

## (Pamornpol Jinachitra)

## (to be presented later this morning)
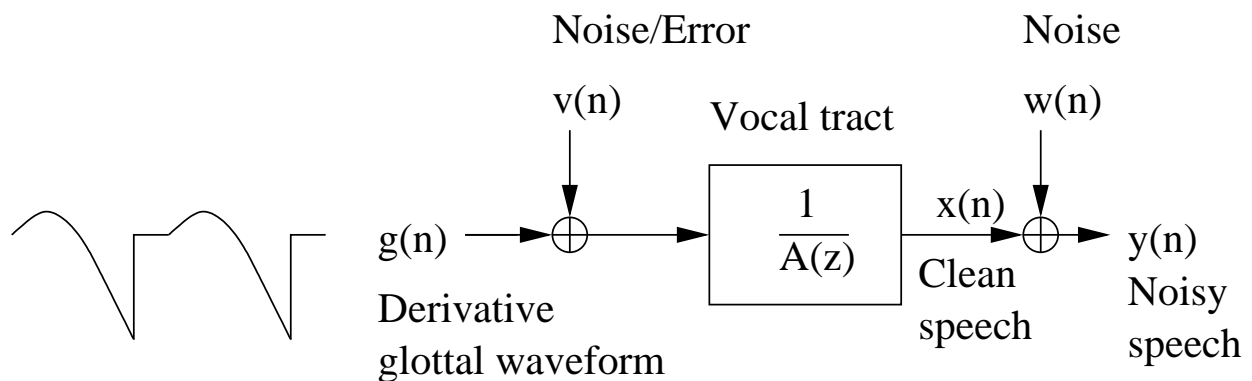


System Diagram

- Parametric source-filter model of voice + noise

- State-space framework with derivative glottal waveform as input and A model for dynamics

- Jointly estimate AR parameters and glottal source parameters using EM algorithm with Kalman smoothing

- Reconstruct a clean voice using Kelly-Lochbaum and estimated parameters

# Karplus-Strong Algorithm

Output $\;y^{+}(n)$    | $N$ samples delay |    $y^{+}(n\text{-}N)$

$+$    1/2

1/2    $z^{-1}$
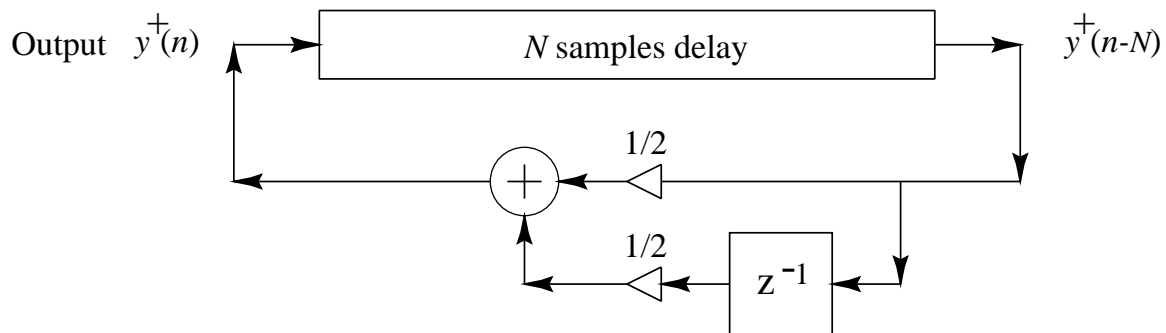
- Discovered (1978) as a self-modifying wavetable synthesis algorithm

- "Vintage" 8-bit sound examples:

  - Original Plucked String: (AIFF) (MP3)
  - Drum: (AIFF) (MP3)
  - Stretched Drum: (AIFF) (MP3)

- STK Plucked String: (WAV) (MP3)

  - Plucked String 1: (WAV) (MP3)
  - Plucked String 2: (WAV) (MP3)
  - Plucked String 3: (WAV) (MP3)
  - Plucked String 4: (WAV) (MP3)

# Interpretations of the Karplus-Strong Algorithm

The Karplus-Strong structure can be *interpreted* as a

- *pitch prediction filter* from the Codebook-Excited Linear Prediction (CELP) standard (*periodic LPC synthesis*)

- *feedback comb filter* with *lowpassed feedback* used earlier by James A. Moorer for recursively modeling *wall-to-wall echoes* ("About This Reverberation Business")

- simplified digital waveguide model

# Digital Waveguide Models

A *lossless digital waveguide* $\triangleq$ *bidirectional delay line* at some *wave impedance* $R$:
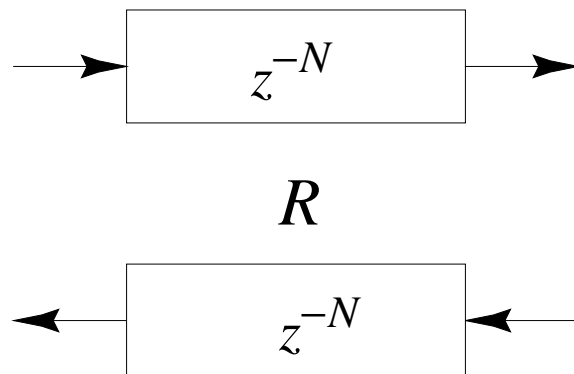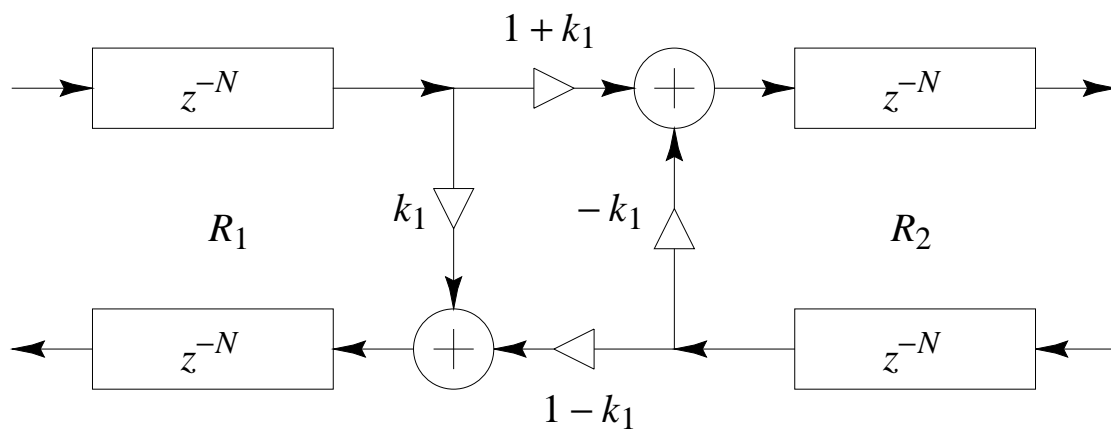


Useful for efficient models of

- strings

- bores

- plane waves

- conical waves

# Signal Scattering

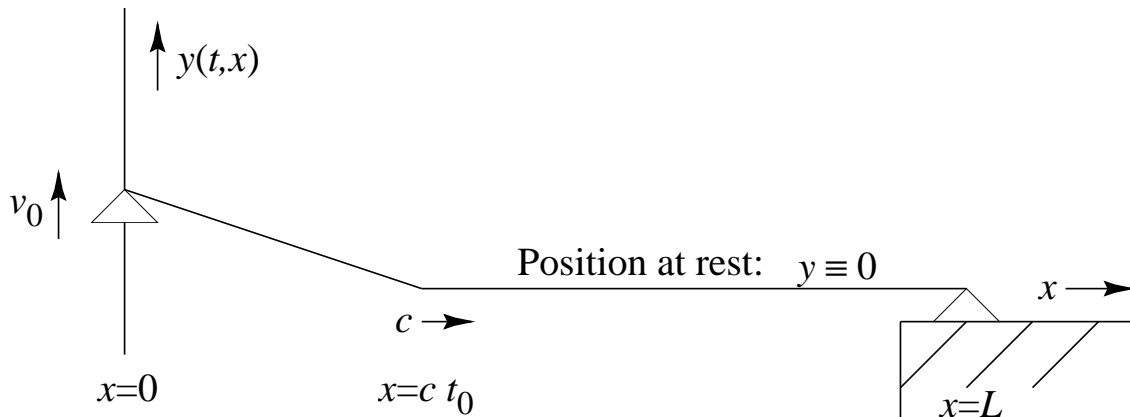Signal *scattering* is caused by a *change* in wave impedance $R$:

$$k_1 = \frac{R_2 - R_1}{R_2 + R_1}$$



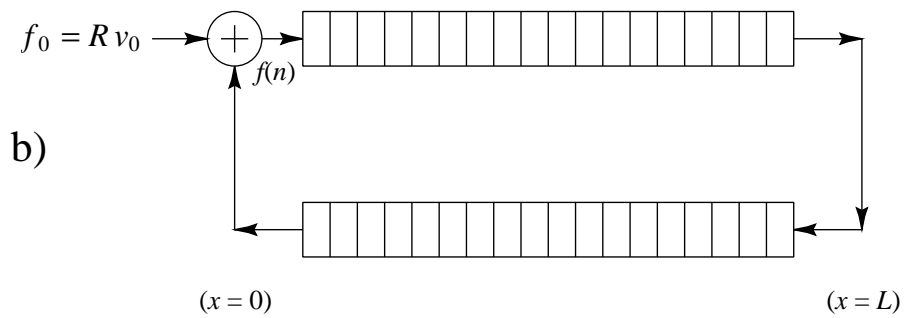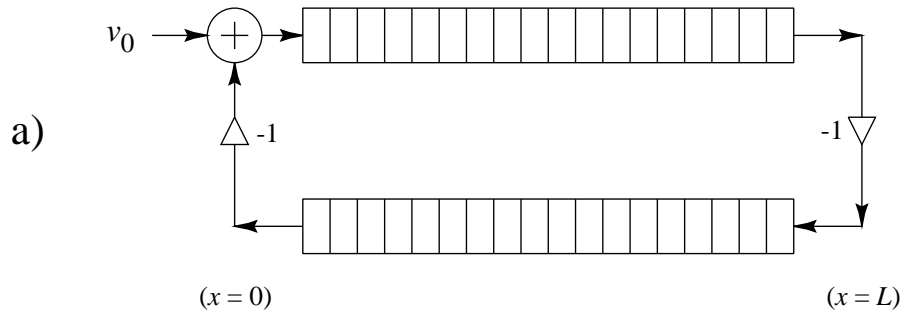If the wave impedance changes *every sample*, the Kelly-Lochbaum vocal-tract model results.

# Moving Termination: Ideal String



Moving rigid termination for an ideal string.

- Left endpoint moved at velocity $v_0$

- External force $f_0 = Rv_0$

- $R = \sqrt{K\epsilon}$ is the *wave impedance* (for transverse waves)

- Relevant to *bowed strings* (when bow pulls string)

- String moves with speed $v_0$ or $0$ only

- String is always one or two straight segments

- "Helmholtz corner" (slope discontinuity) shuttles back and forth at speed $c = \sqrt{K/\epsilon}$
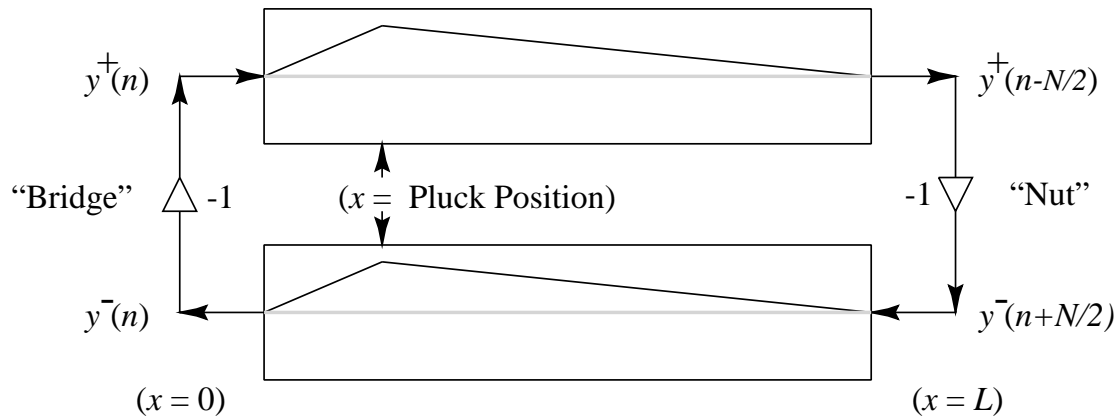
# Digital Waveguide "Equivalent Circuits"

a)

$v_0$

-1

-1

$(x = 0)$  $(x = L)$

b)

$f_0 = R\,v_0$

$f(n)$

$(x = 0)$  $(x = L)$

a) Velocity waves.
b) Force waves.

## Animation:
`http://ccrma.stanford.edu/~jos/swgt/movet.html`

# Ideal Plucked String (Displacement Waves)



$y^+(n)$     $y^+(n\text{-}N/2)$

"Bridge"   -1     ($x =$ Pluck Position)     -1   "Nut"

$y^-(n)$     $y^-(n+N/2)$

($x = 0$)     ($x = L$)

- Load each delay line with *half* of initial string displacement

- Sum of upper and lower delay lines $=$ string displacement
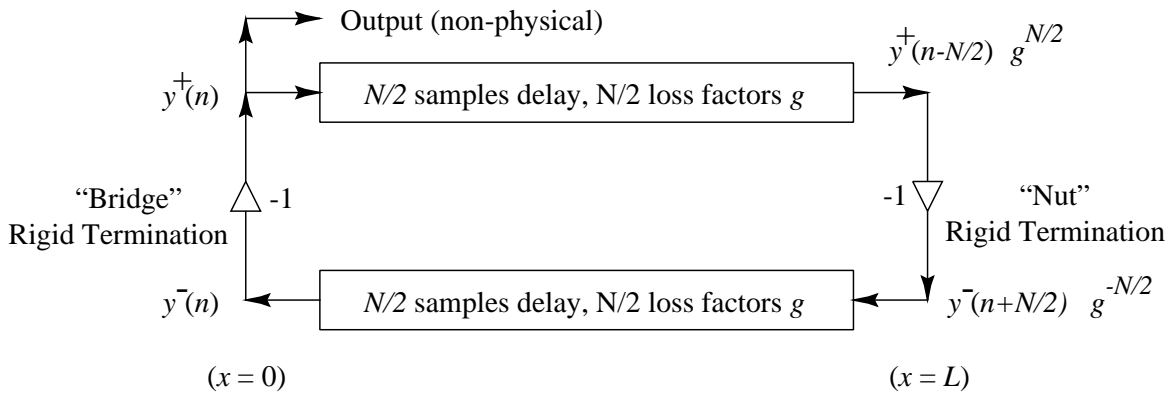
# Ideal Struck String (Velocity Waves)
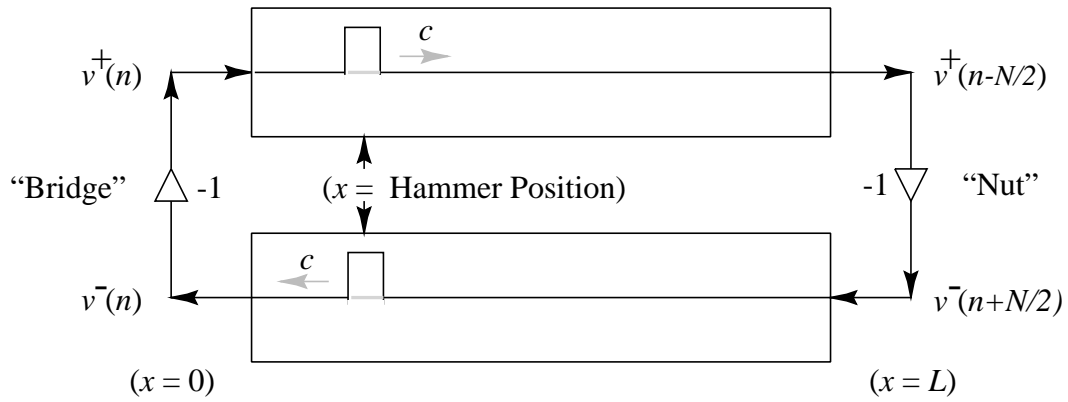


Hammer strike = *momentum transfer* = velocity step:

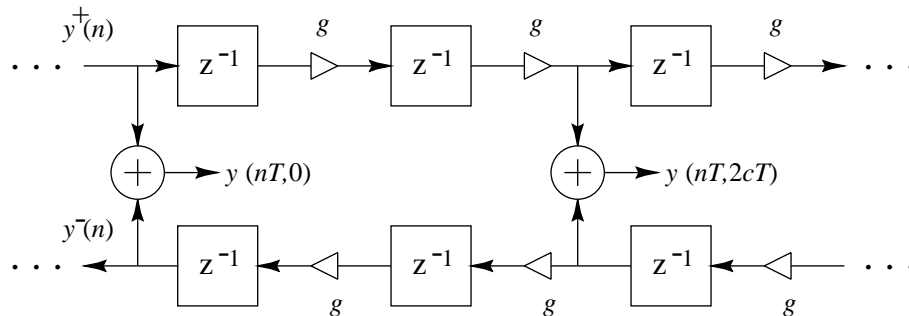$$m_h v_h(0-) = (m_h + m_s)v_s(0+)$$

# Digital Waveguide Interpretation of the Karplus-Strong Algorithm
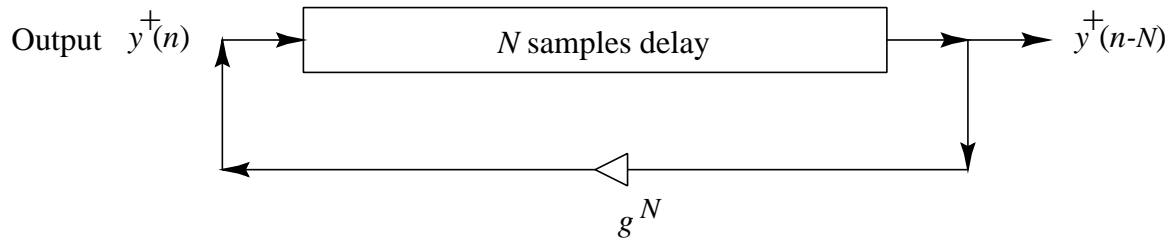
Begin with an ideal *damped* string model:



- Rigidly terminated string with distributed "resistive" losses (force $\propto$ velocity)

- Sampled wave-equation solution yields $N$ loss factors $g$ embedded between the delay-line elements:



- Note that loss factors $g$ *commute* with delay elements

# Equivalent System: Gain Elements Commuted

Output  $y^+(n)$ → [ $N$ samples delay ] → $y^+(n\text{-}N)$

$g^N$

All $N$ loss factors $g$ have been "pushed" through delay elements and combined at a *single* point.

# Computational Savings

- $f_s = 50\text{kHz}, f_1 = 100Hz \Rightarrow \text{delay} = 500$

- Multiplies reduced by two orders of magnitude

- Input-output transfer function unchanged

- Round-off errors reduced

# Frequency-Dependent Damping

- Loss factors $g$ should really be digital filters

- Gains in nature typically decrease with frequency

- Loop gain may not exceed $1$ (for stability)

- Such filters *also commute* with delay elements (LTI)

- Typically only *one* gain filter used per loop

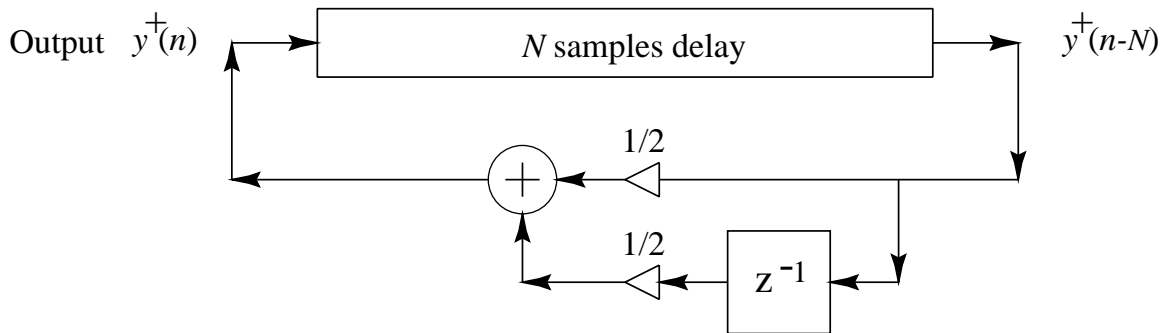## Simplest Frequency-Dependent Loop Filter

$$\boxed{\hat{G}(z) = b_0 + b_1 z^{-1}}$$

- Uniform delay $\Rightarrow b_0 = b_1$ ($\Rightarrow$ delay $= 1/2$ sample)

- Zero damping at dc $\Rightarrow b_0 + b_1 = 1$
$\Rightarrow b_0 = b_1 = 1/2$
$\Rightarrow$

$$\boxed{\hat{G}(e^{j\omega T}) = \cos\left(\omega T/2\right), \quad |\omega| \leq \pi f_s}$$

- This is precisely the Karplus-Strong loop filter!

# Karplus-Strong Algorithm

Output $y^+(n)$ → [ $N$ samples delay ] → $y^+(n\text{-}N)$

1/2 ◁ (+) 1/2 ◁ $z^{-1}$

## Physical Interpretation

- Delay line is initialized with *noise* (random numbers)

- Therefore, assuming a *displacement-wave* simulation:

  - Initial string *displacement* = *sum* of delay-line halves

  - Initial string *velocity* determined by the *difference* of delay-line halves

- The Karplus-Strong "string" is thus *plucked and struck* by random amounts along the *entire length of the string!* (the "splucked string"?)

- Karplus-Strong feedback filter corresponds to the *simplest possible damping filter* for an ideal string

# Extended Karplus-Strong (EKS) Algorithm (Jaffe-Smith 1983)
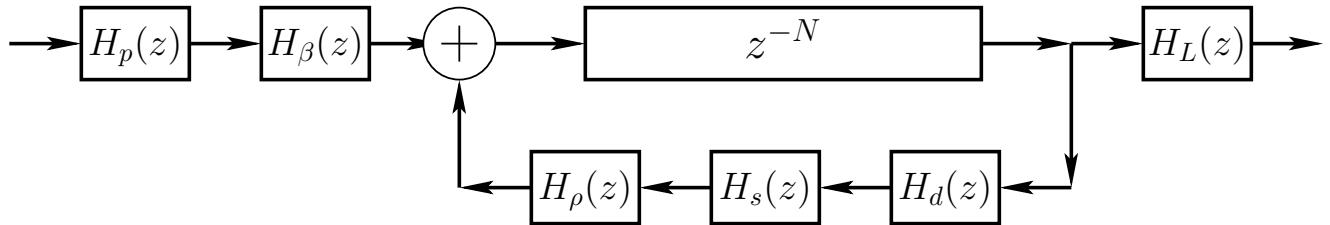


$$N = \text{pitch period } (2\times \text{ string length}) \text{ in samples}$$

$$H_p(z) = \frac{1 - p}{1 - p\,z^{-1}} = \text{pick-direction lowpass filter}$$

$$H_\beta(z) = 1 - z^{-\beta N} = \text{pick-position comb filter, } \beta \in (0, 1)$$

$$H_d(z) = \text{string-damping filter (one/two poles/zeros typical)}$$

$$H_s(z) = \text{string-stiffness allpass filter (several poles and zeros)}$$

$$H_\rho(z) = \frac{\rho(N) - z^{-1}}{1 - \rho(N)\,z^{-1}} = \text{first-order string-tuning allpass filter}$$

$$H_L(z) = \frac{1 - R_L}{1 - R_L\,z^{-1}} = \text{dynamic-level lowpass filter}$$

# EKS Sound Example

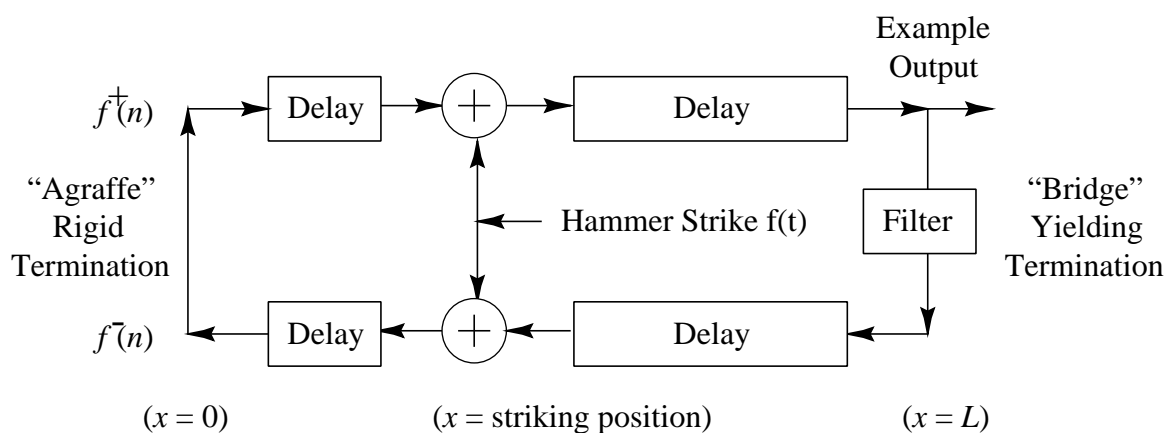Bach A-Minor Concerto—Orchestra Part: (WAV) (MP3)

- Executes in real time on one Motorola DSP56001 (20 MHz clock, 128K SRAM)

- Developed for the NeXT Computer introduction at Davies Symphony Hall, San Francisco, 1989

- Solo violin part was played live by Dan Kobialka of the San Francisco Symphony

# Example EKS Extension

Many of the Karplus-Strong algorithm extensions were based on its *physical interpretation*.

- Originally, transfer-function methods were used.

- Below, we will use digital waveguide methods, which came a couple of years later.

## String Excited Externally at One Point



$$f^+(n) \rightarrow \boxed{\text{Delay}} \rightarrow \oplus \rightarrow \boxed{\text{Delay}} \rightarrow \text{Example Output}$$

"Agraffe" Rigid Termination     Hammer Strike f(t)     Filter     "Bridge" Yielding Termination

$$f^-(n) \leftarrow \boxed{\text{Delay}} \leftarrow \oplus \leftarrow \boxed{\text{Delay}}$$

$(x = 0)$     $(x = \text{striking position})$     $(x = L)$
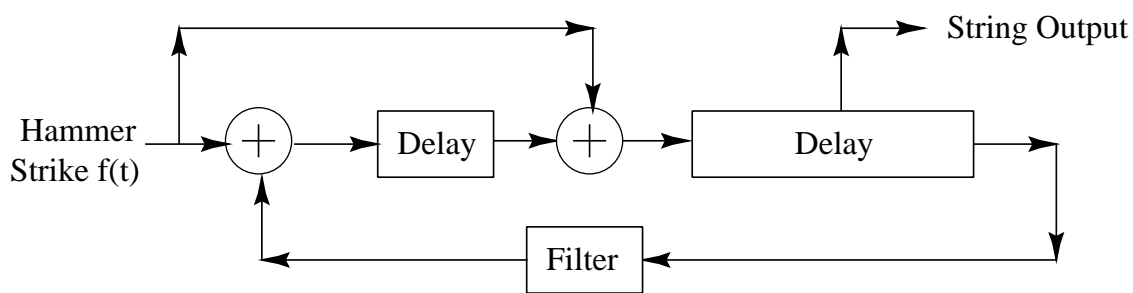
"Waveguide Canonical Form"
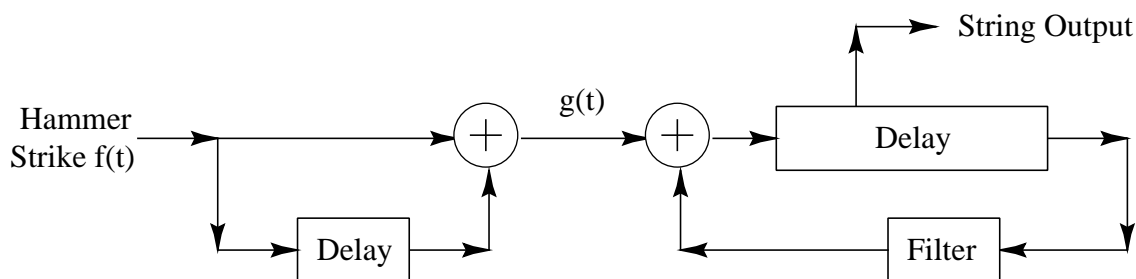
27

# Equivalent System: Delay Consolidation
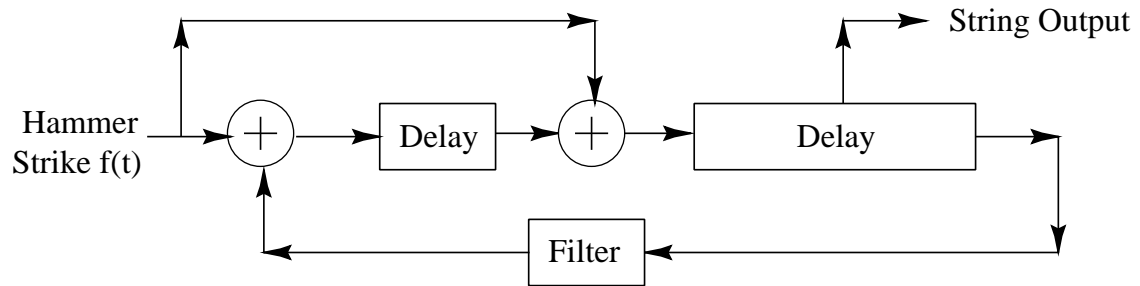


Finally, we "pull out" the comb-filter component:

# EKS "Pick Position" Extension

## Equivalent System: Delay-Lines Consolidated

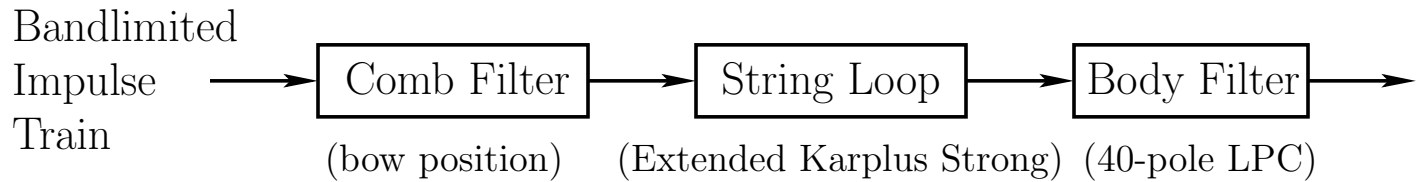

## Equivalent System: Comb Filter Factored Out



- *Excitation Position* controlled by left delay-line length

- *Fundamental Frequency* controlled by right delay-line length

# PLPC Cello (1982)

Bandlimited
Impulse ⟶ | Comb Filter | ⟶ | String Loop | ⟶ | Body Filter | ⟶
Train
        (bow position)   (Extended Karplus Strong)  (40-pole LPC)

- Periodic LPC used to estimate string-loop filter

- Normal LPC used for body model (40 poles)

- Excitation = Bandlimited impulse train:

$$\sum_{k=1}^{K} \cos(k\omega_0 t) = \frac{\sin[(K+1/2)\omega_0 t]}{2\sin(\omega_0 t/2)} - \frac{1}{2}$$

- Bow-position simulation = variable-delay differencing comb filter (direct from physical interpretation)

- **Sound Example:**
  Moving Bow-Stroke Example: (WAV) (MP3)
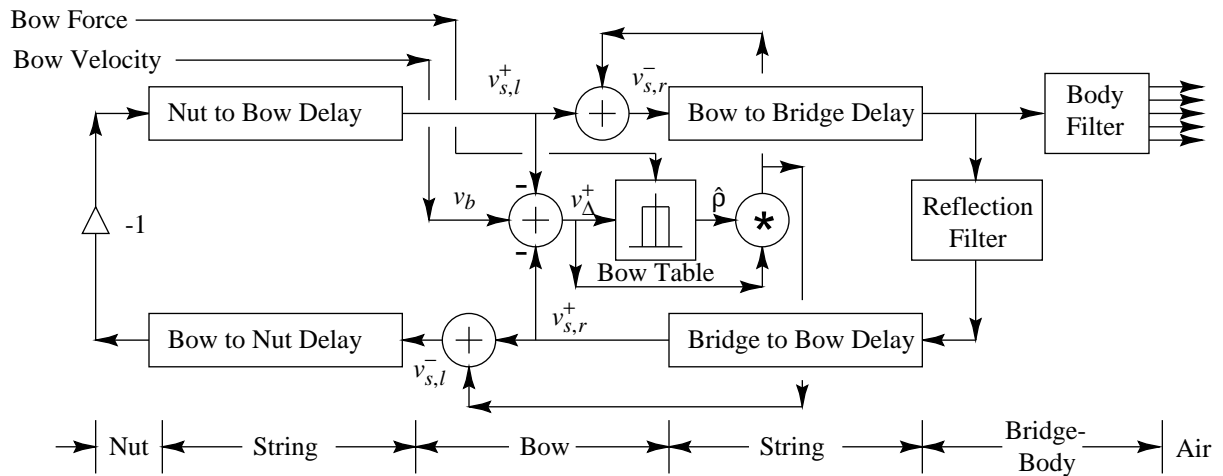  (Bowing point moves toward the "bridge")

# Single-Reed Instruments (1986)
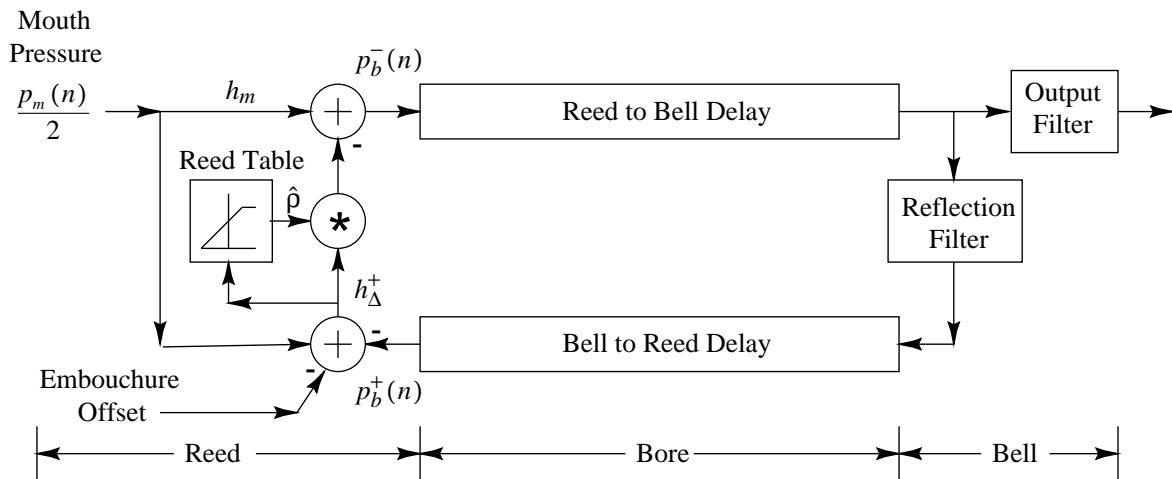


## Sound Examples

- STK Clarinet: (WAV) (MP3)
- Staccato Systems Slide Flute (based on STK flute): (WAV) (MP3)
- Yamaha VL1 Shakuhachi: (WAV) (MP3)
- Yamaha VL1 Oboe and Bassoon: (WAV) (MP3)
- VL1 Tenor Saxophone: (WAV) (MP3)
- `Google` search: *STK clarinet*
- Synthesis Tool Kit (STK) by Perry Cook, Gary Scavone, and others, distributed by CCRMA:
  `http://ccrma.stanford.edu/CCRMA/Software/STK/`

# Bowed Strings (1986)



The block diagram contains the following labeled elements:

- Bow Force → (line into junction)
- Bow Velocity → (line into junction)
- Nut to Bow Delay
- $v^+_{s,l}$
- $v^-_{s,r}$
- Bow to Bridge Delay
- Body Filter
- $-1$
- $v_b$
- $v^+_\Delta$
- $\hat\rho$
- $*$
- Reflection Filter
- Bow Table
- Bow to Nut Delay
- $v^+_{s,r}$
- $v^-_{s,l}$
- Bridge to Bow Delay

Axis labels: Nut — String — Bow — String — Bridge-Body — Air

- **Reflection filter summarizes all losses per period (due to bridge, bow, finger, etc.)**

- **Bow-string junction = *memoryless* lookup table (or segmented polynomial) ⇒ no thermodynamic model in this version**

- **Bow-hair dynamics neglected**

- **Finite bow width neglected**

# Bowed String Sound Examples

Cello sound examples by Stanford EE graduate student **Peder Larson** using the Synthesis Tool Kit (STK) by Perry Cook and Gary Scavone:

- STK Bowed class, no modifications: (WAV) (MP3)

- Hyperbolic Bow-String Junction, including: (WAV) (MP3)

  - Torsional waves: (WAV) (MP3)
  - Finite Bow Width: (WAV) (MP3)
  - Finite Bow Width and Torsional waves: (WAV) (MP3)
  - Finite Bow Width and Body Filter: (WAV) (MP3)
  - Torsional waves and Body Filter: (WAV) (MP3)
  - Finite Bow Width,Torsional waves, Body Filter: (WAV) (MP3)
  - String Dispersion (Stiffness): (WAV) (MP3)
  - Including Torsional waves and dispersion: (WAV) (MP3)
  - Finite Bow Width and Dispersion: (WAV) (MP3)
  - Finite Bow Width, Torsional Waves, Dispersion: (WAV) (MP3)

- Finite Bow Width, Body Filter, and Dispersion: (WAV) (MP3)
- Torsional waves, Body Filter, and Dispersion: (WAV) (MP3)
- Same plus Finite Bow Width: (WAV) (MP3)

## Cello Examples Using All Features

- Stacatto Notes: (WAV) (MP3)
- Bach's First Suite for Unaccompanied Cello: (WAV) (MP3)

Staccato notes created with short strokes of high bow pressure (like a bouncing bow)
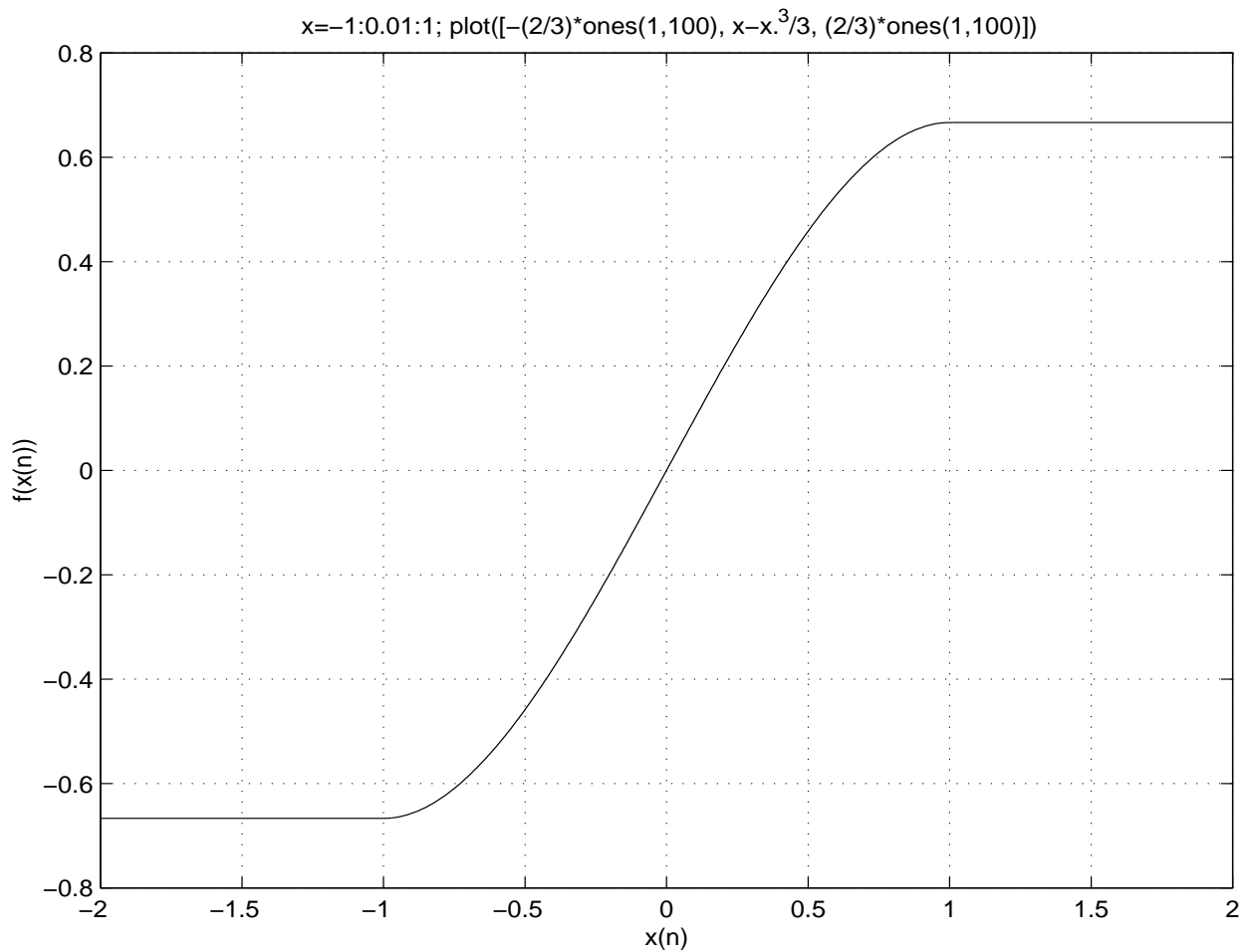
## Without Dispersion

- Stacatto notes: (WAV) (MP3)
- Bach's First Suite for Unaccompanied Cello: (WAV) (MP3)

# Nonlinear "Overdrive"

## Soft Clipper

$$
f(x) = \begin{cases} -\frac{2}{3}, & x \leq -1 \\ x - \frac{x^3}{3}, & -1 \leq x \leq 1 \\ \frac{2}{3}, & x \geq 1 \end{cases}
$$

x=−1:0.01:1; plot([−(2/3)*ones(1,100), x−x.³/3, (2/3)*ones(1,100)])

# Amplifier Distortion + Amplifier Feedback

## Sullivan 1990
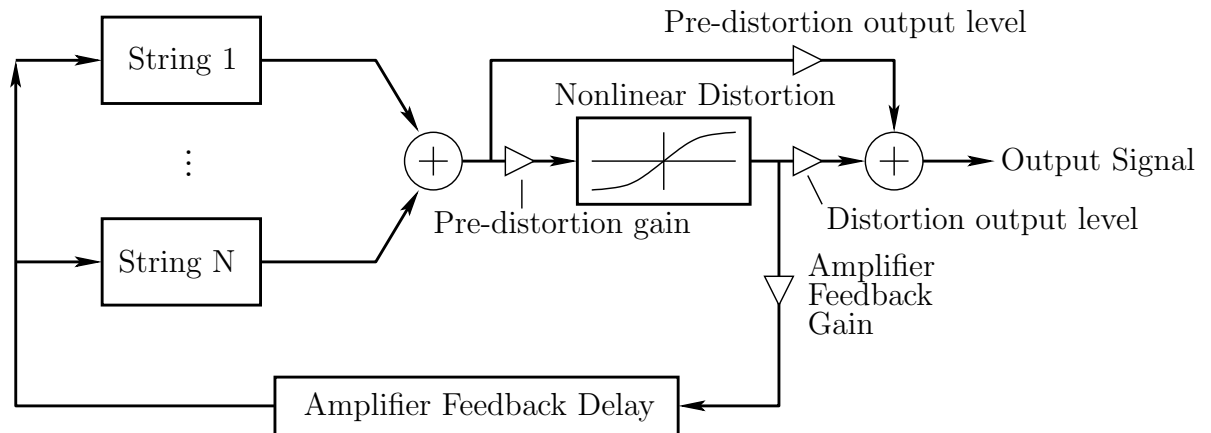


Distortion output signal often further filtered by an
*amplifier cabinet filter*, representing speaker cabinet,
driver responses, etc.

## Sound Examples

- Distortion Guitar: (WAV) (MP3)
- Amplifier Feedback 1: (WAV) (MP3)
- Amplifier Feedback 2: (WAV) (MP3)

# Commuted Synthesis of Acoustic Strings (1993)

Trigger → | Excitation | →e(t)→ | String | →s(t)→ | **Resonator** | →y(t)→ Output

Schematic diagram of a stringed musical instrument.

Trigger → | Excitation | → | **Resonator** | → | String | → Output

Equivalent diagram in the linear, time-invariant case.

Trigger → | **Aggregate Excitation** | →a(t)→ | String | →x(t)→ Output
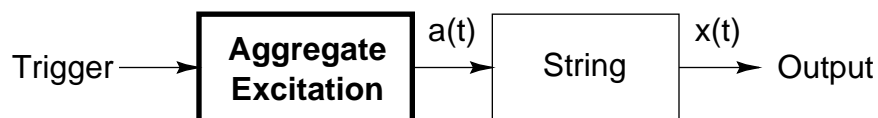
Use of an aggregate excitation given by the convolution of original excitation with the resonator impulse response.

s(t) → | Bridge Coupling | → | Guitar Body | → | Air Absorption | → | Room Response | →y(t)→ Output
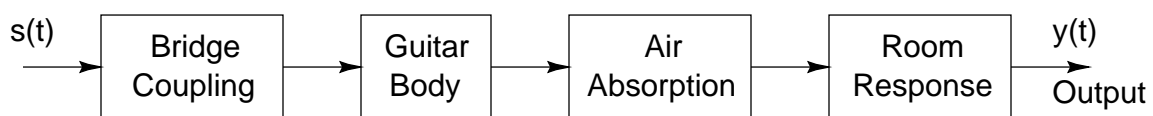
Possible components of a guitar resonator.

# Commuted Synthesis Sound Examples

## Acoustic Guitar

- Bach Prelude in E Major: (AIFF) (MP3)
- Bach Loure in E Major: (AIFF) (MP3)

Virtual performance by Dr. Mikael Laurson[1], Sibelius Institute
Virtual guitar by Helsinki University of Technology, Acoustics Lab[2]

## Electric Guitar (Pick-Ups and/or Body-Model Added)

- Example 1: (WAV) (MP3)
- Example 2: (WAV) (MP3)
- Example 3: (WAV) (MP3)
- Virtual "wah-wah pedal": (WAV) (MP3)

Stanford Sondius Project
Staccato Systems, Inc.

## STK Mandolin

- STK Mandolin 1: (WAV) (MP3)
- STK Mandolin 2: (WAV) (MP3)

---

[1] http://www2.siba.fi/soundingscore/MikaelsHomePage/MikaelsHomepage.html
[2] http://www.acoustics.hut.fi/

# Commuted Synthesis of Linearized Violin

**a)**

Amplitude(n) → [ Impulse Train ] —e(n)→ [ String ] —s(n)→ [ Resonator ] —x(n)→ Output
Frequency(n) →

**b)**

Amplitude(n) → [ Impulse Train ] —e(n)→ [ Resonator ] —a(n)→ [ String ] —x(n)→ Output
Frequency(n) →

**c)**

Amplitude(n) → [ Impulse-Response Train ] —a(n)→ [ String ] —x(n)→ Output
Frequency(n) →
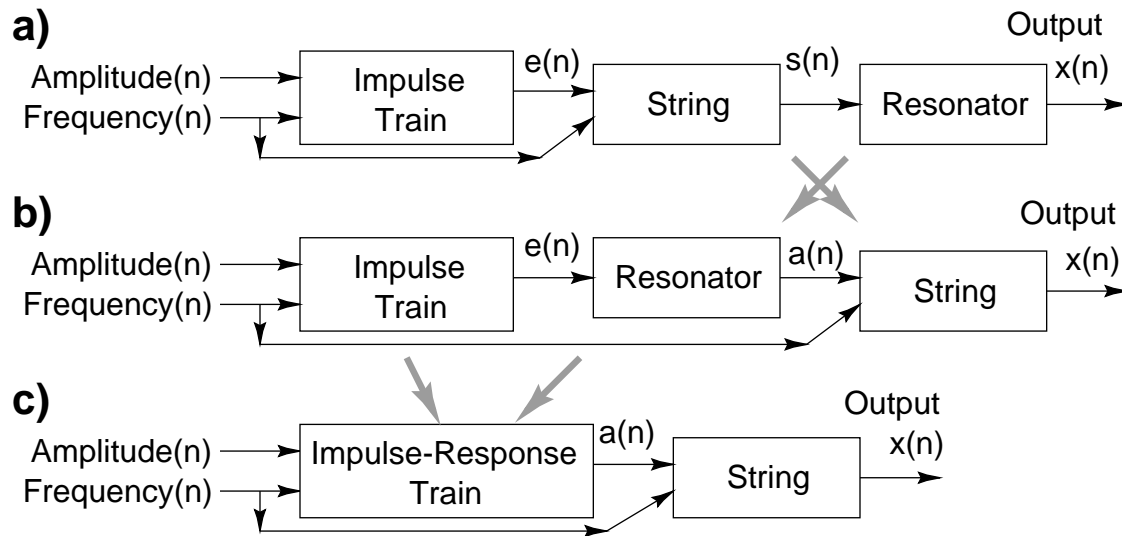
- Assumes *ideal Helmholtz motion* of string

- Sound Examples:

  - Double Bass: (WAV) (MP3)
  - Cello: (WAV) (MP3)
  - Viola 1: (WAV) (MP3)
  - Viola 2: (WAV) (MP3)
  - Violin 1: (WAV) (MP3)
  - Violin 2: (WAV) (MP3)
  - Ensemble: (WAV) (MP3)

Stanford Sondius Project
Staccato Systems, Inc.

# Commuted Piano Synthesis (1995)

Hammer-string interaction pulses (force):
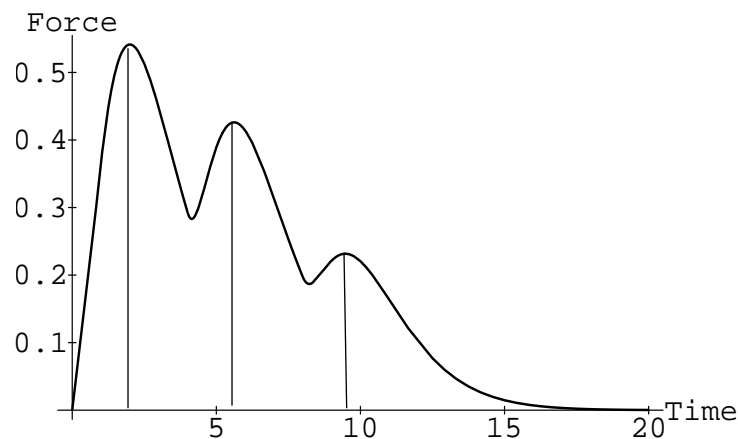


Vertical lines specify three *impulses* which will drive one to three *pulse-synthesis filters*

- Hammer = *mass* covered by *nonlinear spring* ("felt")

- String looks like a *resistor* upon initial impact

- Second and third pulses caused by *reflections* from agraffe (number depends on key number and hammer velocity)

- Pulses taller and thinner when hammer-velocity larger

# Synthesis of Hammer-String Interaction Pulse

Impulse               Lowpass Filter              Impulse Response

Time                                   Time
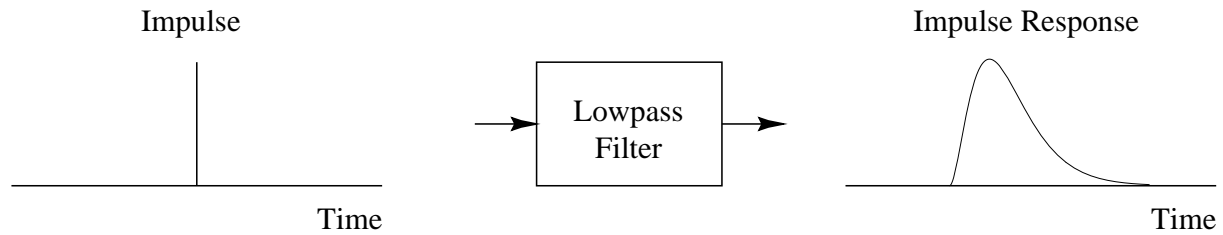
- Faster collisions correspond to *narrower* pulses (*nonlinear filter*)

- For a *given velocity*, filter is linear time-invariant

- Piano is "linearized" for each hammer velocity

# Multiple Hammer-String Interaction Pulses

Superimpose several individual pulses:
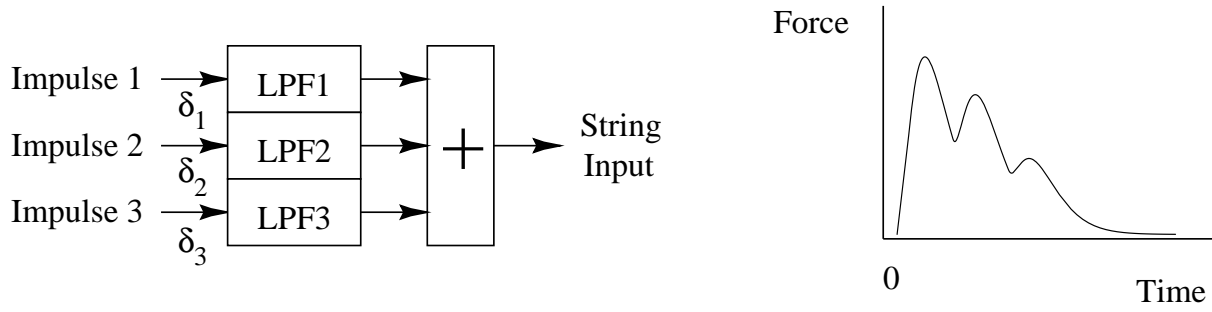


As impulse amplitude grows (faster hammer strike), output pulses become *taller and thinner*, showing less overlap.

# Complete Piano Model

## Natural Ordering:
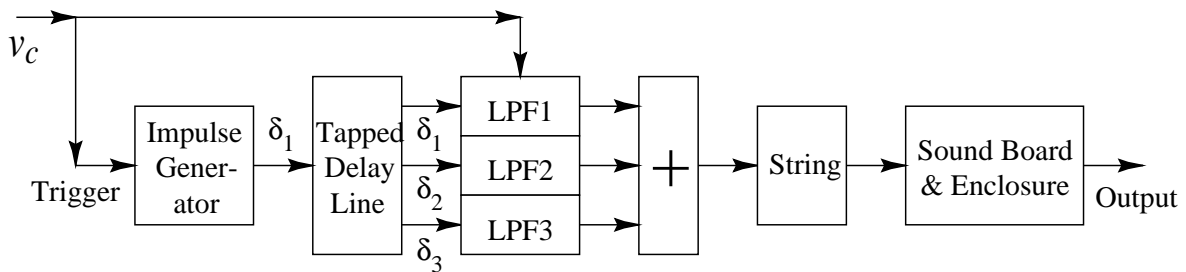


## Commuted Ordering:
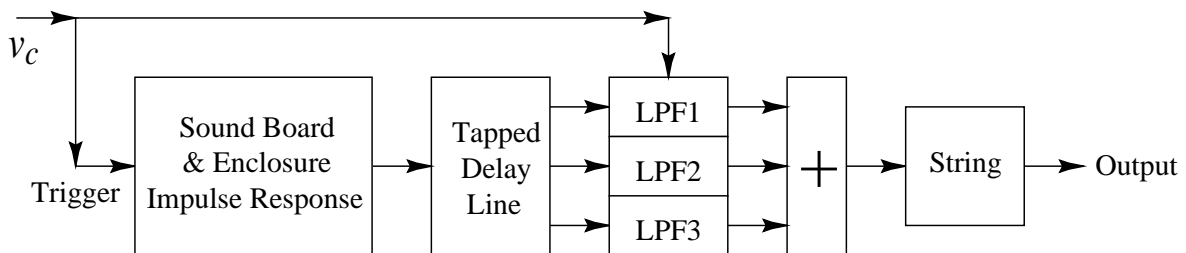


- Soundboard and enclosure are *commuted*

- Only need a stored recording of their *impulse response*

- An enormous digital filter is otherwise required

# Sound Examples

## Piano and Harpsichord:

- Piano: (WAV) (MP3)
- Harpsichord 1: (WAV) (MP3)
- Harpsichord 2: (WAV) (MP3)

Stanford Sondius Project
Staccato Systems, Inc.

## More Recent Harpsichord:

- Harpsichord Soundboard Hammer-Response: (WAV) (MP3)
- Musical Commuted Harpsichord Example: (WAV) (MP3)

Vesa Välimäki, Henri Penttinen, Jonte Knif, Mikael Laurson, and Cumhur Erkut
"Sound Synthesis of the Harpsichord Using a Computationally Efficient Physical Model", JASP-2004
http://www.acoustics.hut.fi/publications/papers/jasp-harpsy/
Google search: *Harpsichord Sound Synthesis*

# Virtual Analog Synthesis

Most "Virtual Analog" synthesizers try to emulate some version of the MiniMoog or MemoryMoog synthesizers, because of their popularity. These classic synths were designed by the analog-synth pioneer Robert Moog.



Early Examples of Virtual Analog Synths:

- Nordlead "Virtual Analog Synthesizer"
- Roland "Analog Modeling Synthesis"
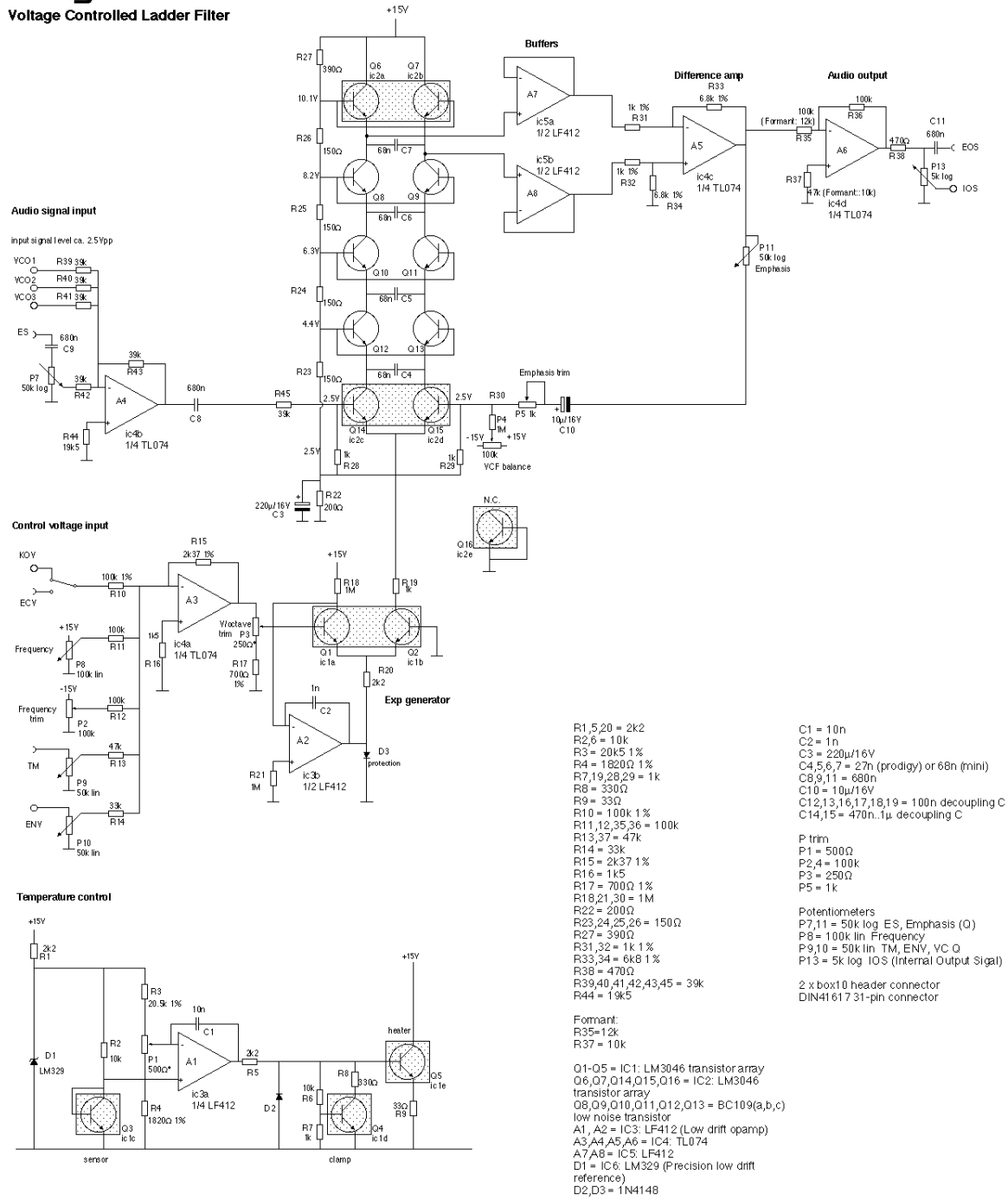- Yamaha "Analog Physical Modeling" (AN-1)

**Design goal:** Emulate the Moog Voltage Controlled Filter (VCF), due its popularity and excellent properties.

# Moog VCF Ladder (1966)

# moog
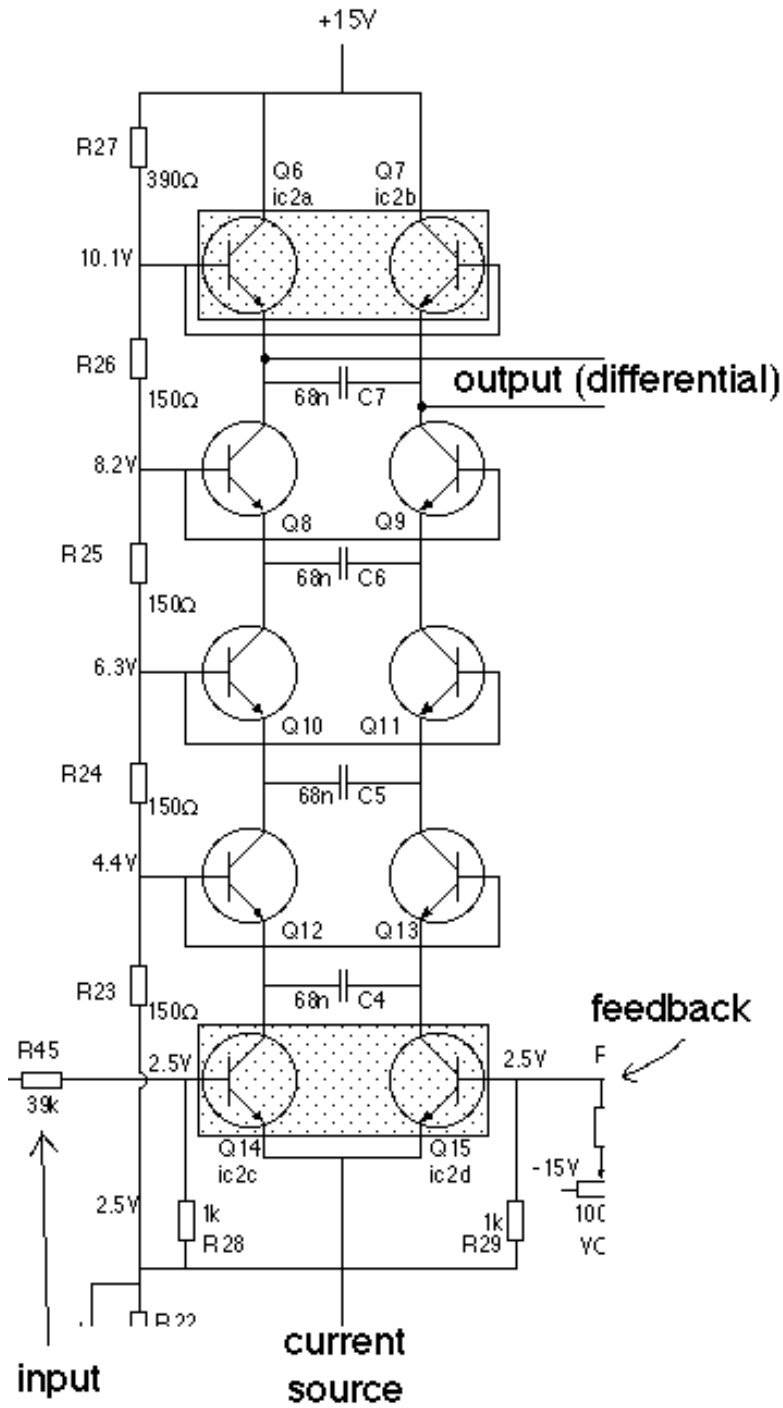
**Voltage Controlled Ladder Filter**

Buffers

Difference amp

Audio output

Audio signal input

input signal level ca. 2.5Vpp
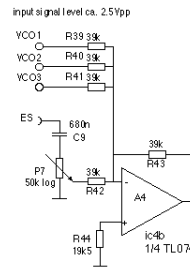
Control voltage input

Exp generator
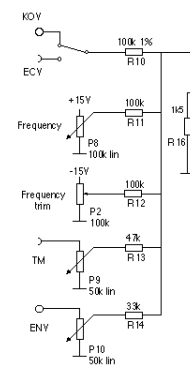
Temperature control
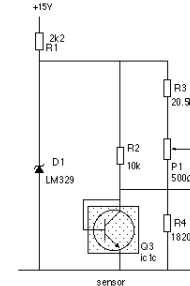
R1,5,20 = 2k2
R2,6 = 10k
R3 = 20k5 1%
R4 = 1820Ω 1%
R7,19,28,29 = 1k
R8 = 330Ω
R9 = 33Ω
R10 = 100k 1%
R11,12,35,36 = 100k
R13,37 = 47k
R14 = 33k
R15 = 2k37 1%
R16 = 1k5
R17 = 700Ω 1%
R18,21,30 = 1M
R22 = 200Ω
R23,24,25,26 = 150Ω
R27 = 390Ω
R31,32 = 1k 1%
R33,34 = 6k8 1%
R38 = 470Ω
R39,40,41,42,43,45 = 39k
R44 = 19k5

Formant:
R35 = 12k
R37 = 10k

C1 = 10n
C2 = 1n
C3 = 220μ/16V
C4,5,6,7 = 27n (prodigy) or 68n (mini)
C8,9,11 = 680n
C10 = 10μ/16V
C12,13,16,17,18,19 = 100n decoupling C
C14,15 = 470n..1μ decoupling C

P trim
P1 = 500Ω
P2,4 = 100k
P3 = 250Ω
P5 = 1k

Potentiometers
P7,11 = 50k log ES, Emphasis (Q)
P8 = 100k lin Frequency
P9,10 = 50k lin TM, ENV, VC Q
P13 = 5k log IOS (Internal Output Sigal)

2 x box10 header connector
DIN41617 31-pin connector

Q1-Q5 = IC1: LM3046 transistor array
Q6,Q7,Q14,Q15,Q16 = IC2: LM3046
transistor array
Q8,Q9,Q10,Q11,Q12,Q13 = BC109(a,b,c)
low noise transistor
A1, A2 = IC3: LF412 (Low drift opamp)
A3,A4,A5,A6 = IC4: TL074
A7,A8 = IC5: LF412
D1 = IC6: LM329 (Precision low drift
reference)
D2,D3 = 1N4148

• All resistors recommended metal film
• Cermet trimmers recommended (especially P2 and P3)
• Capacitors MKM or MKF recommended

• The exp generator is inspired by the Elektor Formant VCO's exp generator
• Temperature control as in National Semiconductor Application Note AN-299 with added temperature adjustment.
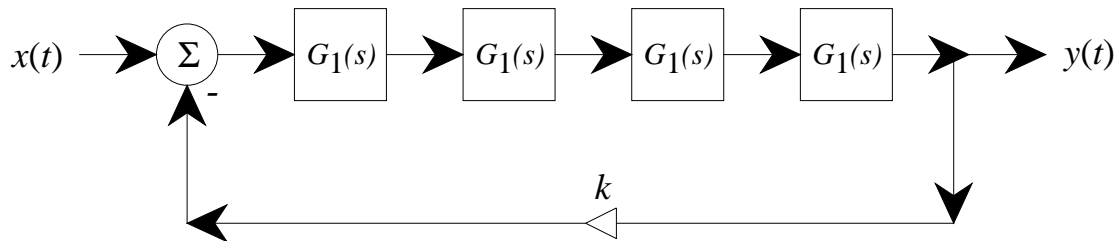• Pin-compatible with the Formant regular VCF

Revision 1.2
Last change: June 10th 1996

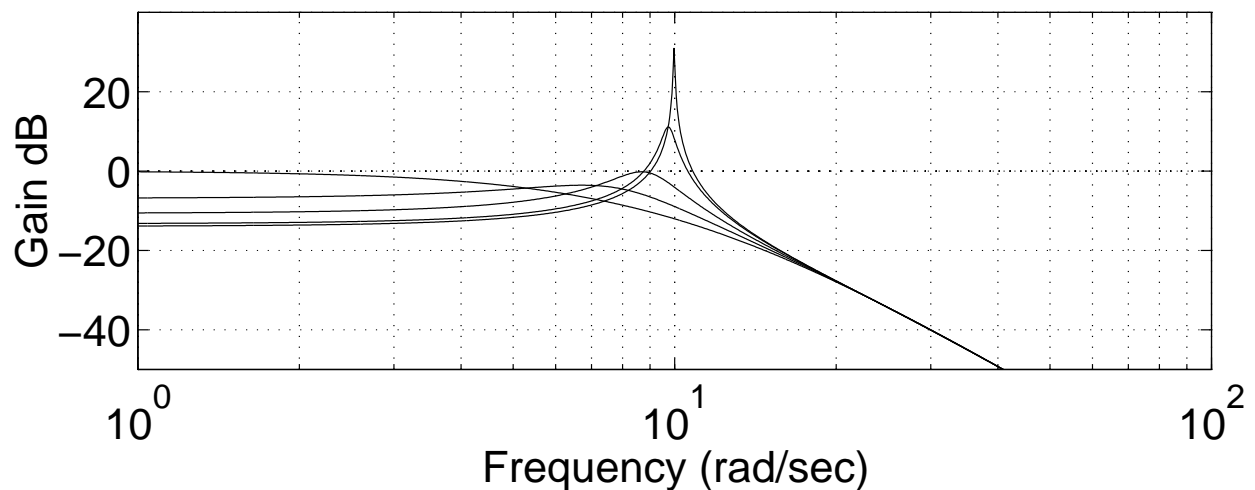Many thanks to Don Tillman and Barry Klein for invaluable discussions.

# The Moog VCF (1966)

**Structure:** four identical one-poles in series with a feedback loop:
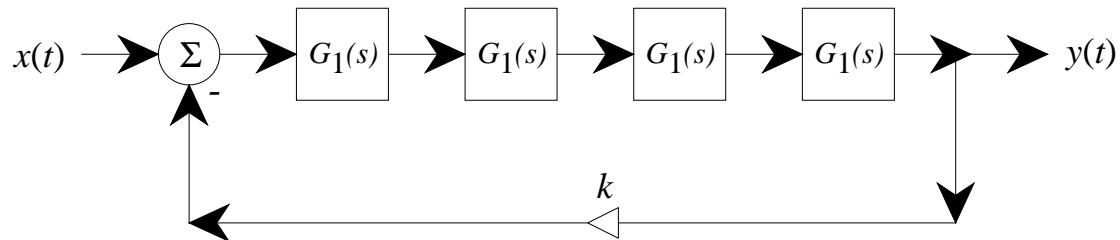


This implements a voltage-controllable four-pole filter:



In the Moog analog ladder (the "Moog 904A"), the one-pole filters consist of the capacitors and the AC resistance of the transistors, which is determined by the current source, which is varied to control tuning. See US Patent 3,475,623

# Moog VCF Controls



## Controls

- One-pole pole location: controls *cut-off frequency*

- Feedback gain: controls *resonance*

## Resonance

- The phase of each pole at $s = -a$ is $45°$ when $s = ja$

- At this frequency $(\omega = a)$, the phase through all four filters is $180°$

- The gain of each one-pole at $\omega = a$ is $1/\sqrt{2}$
  $\Rightarrow$ total gain is $1/4$

- Therefore, with a feedback gain of $k = 4$, the loop has *in-phase, positive feedback* at frequency $\omega = a$
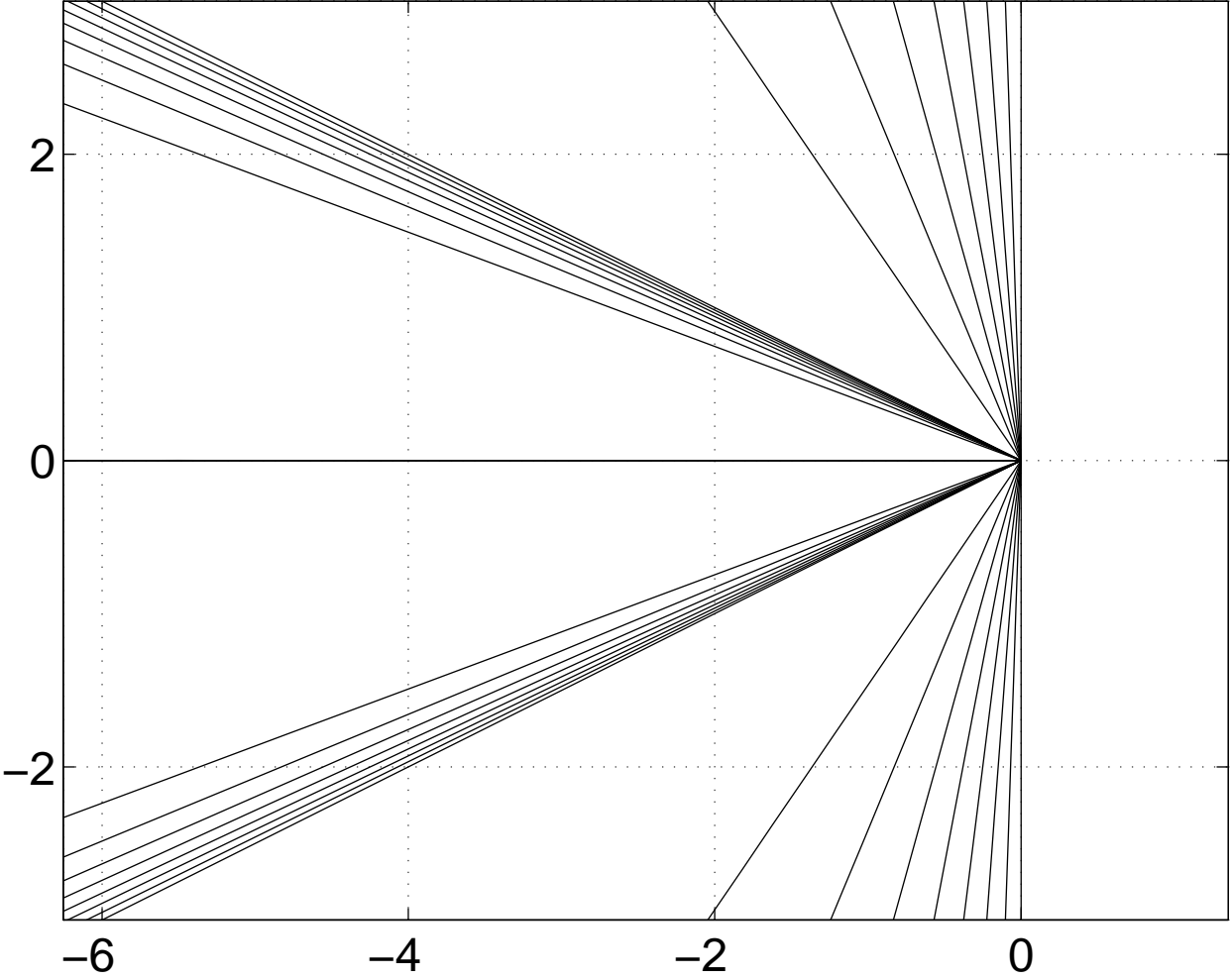
# Moog VCF Root Locus

Some root loci of the continuous-time Moog VCF:

## Root locus as $k$ varies:
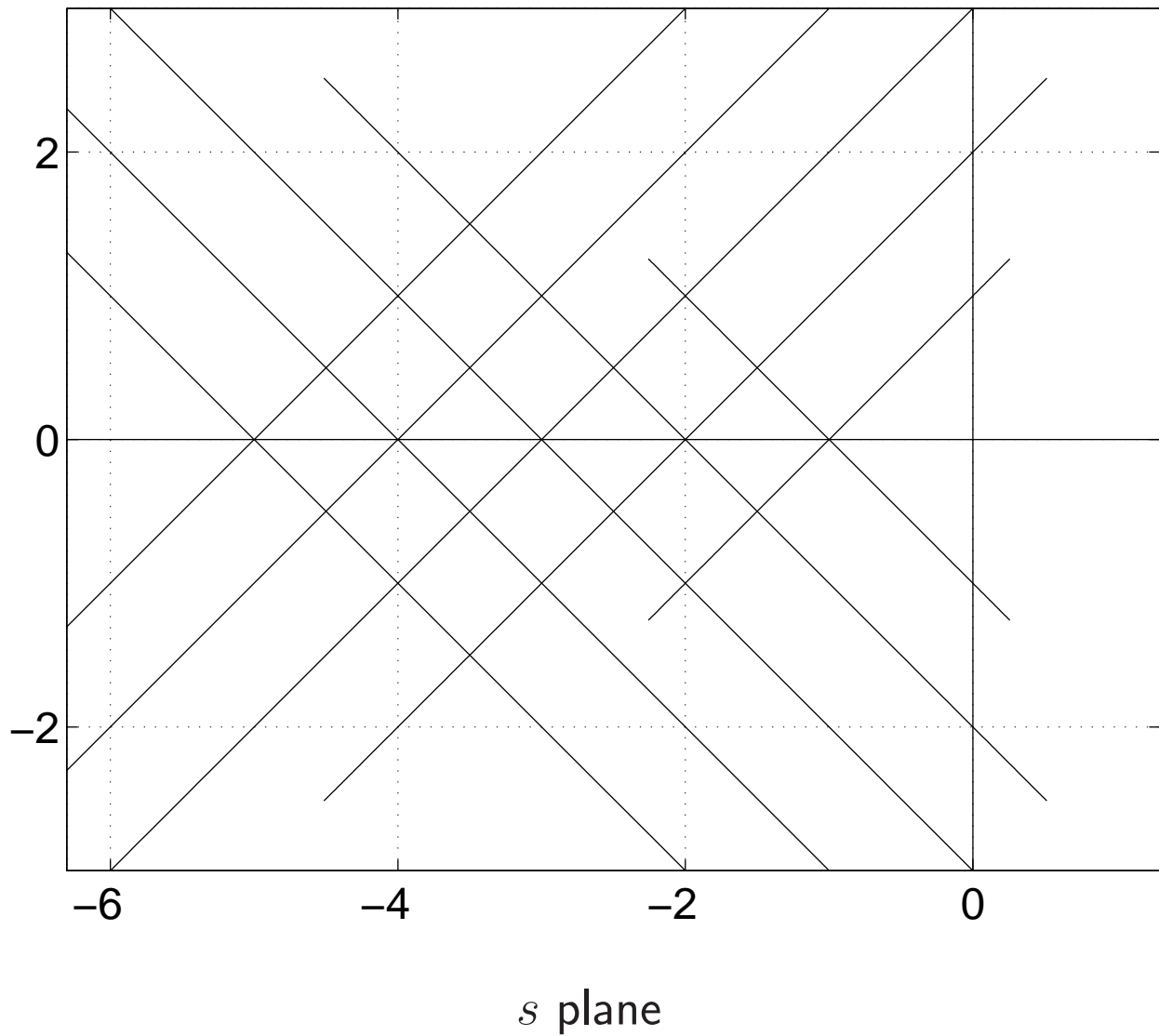


$s$ plane

# Root locus as one-pole pole location ($p$) varies:



$s$ plane

# Features of the Moog VCF

These loci show why the Moog VCF is such a good structure:

- Controls for cut-off and Q are completely orthogonal (constant-$k$ contours are constant-Q contours)

- Controls are simply related to circuit parameters (resonance frequency = open-loop poles)

# Discrete-Time Moog VCF (1996)

Stilson and Smith, ICMC-96

Within the original structure (four one-poles in series with feedback around them), try various transforms from $s$ to $z$:

**Backward difference:** $G(z) = (p + 1)z/(z + p)$

**Bilinear:** $G(z) = 0.5(p + 1)(z + 1)/(z + p)$

- **Problem:** Delay-free loops: These transforms cause the one-poles to have delay-free paths. Adding a delay to the loop changes the structure, so that complete orthogonality is no longer true (before the added delay, the bilinear case *was* orthogonal).

- *Separation tables* become necessary: The table we will use, in the bilinear-transform case, is a one-dimensional table that contains the feedback gain necessary to make the filter unstable, indexed by $p$. We scale $k$ by this table ($k = \text{table}(p)k_{in}$).

- We would like to find a structure for which a separation table is unnecessary.
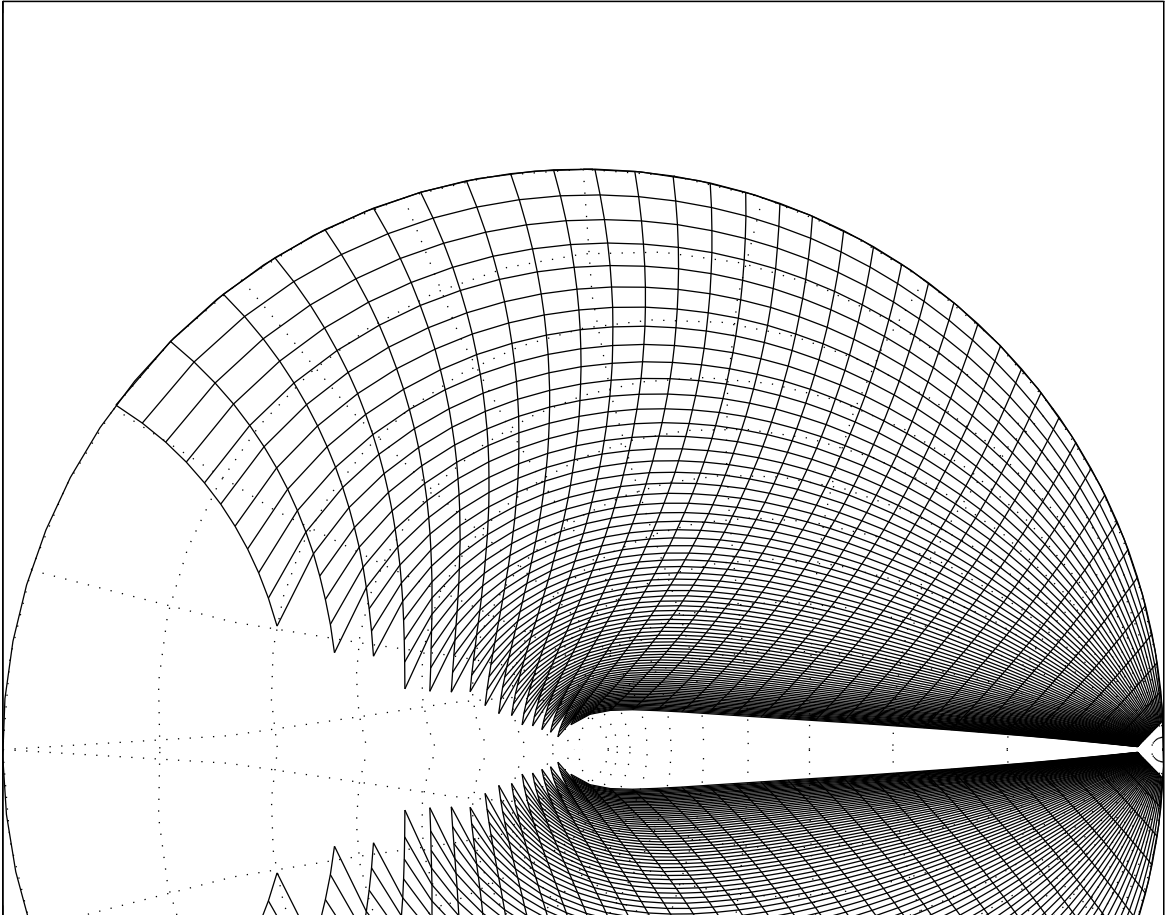
# Separation Table

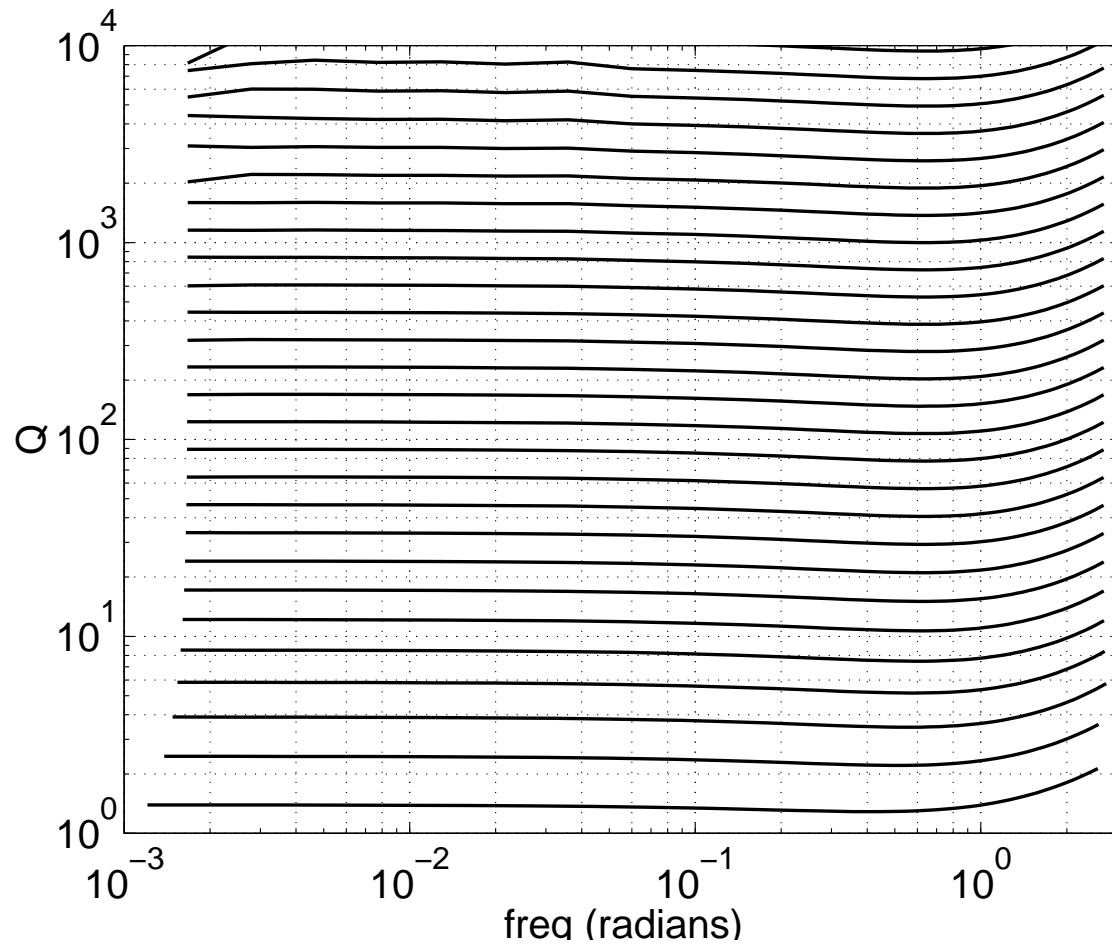For the bilinear case, the separation table is the top curve:
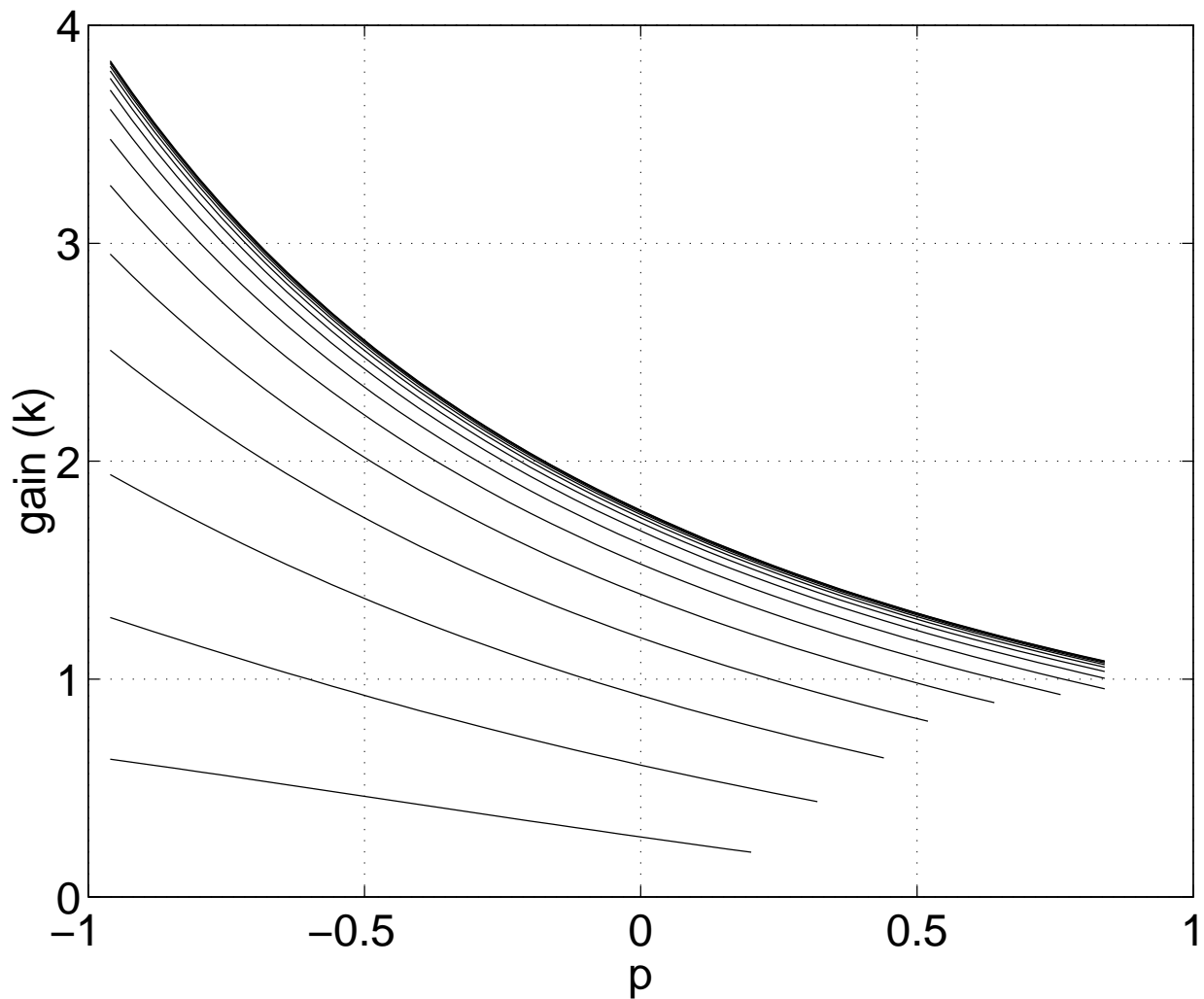


The other curves show gains for various values of Q
(the top is infinite-Q).

# Root Locus, Bilinear Transform Case
## (Using Separation Table)



Each curve generated by sweeping $p$ with a fixed $k$

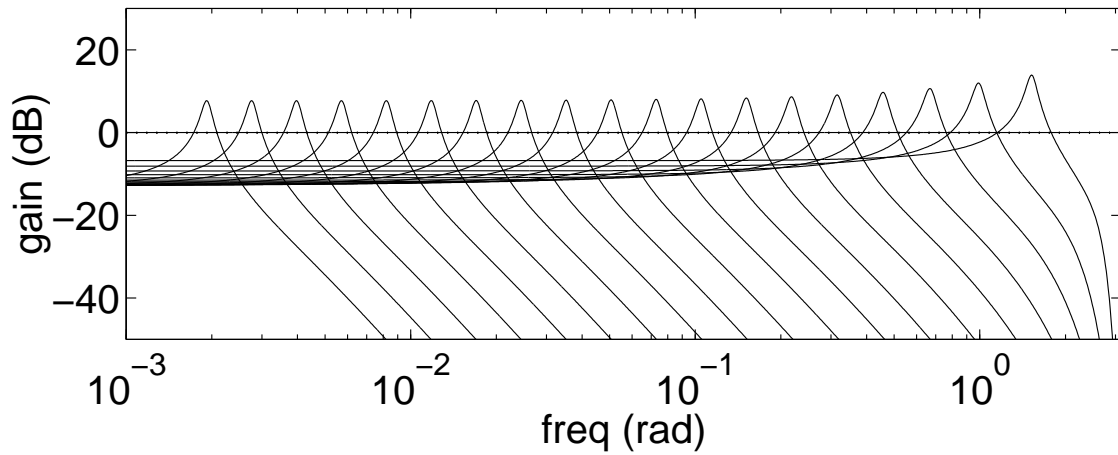# Q versus Cut-Off Frequency, Bilinear Transform, With Separation Table



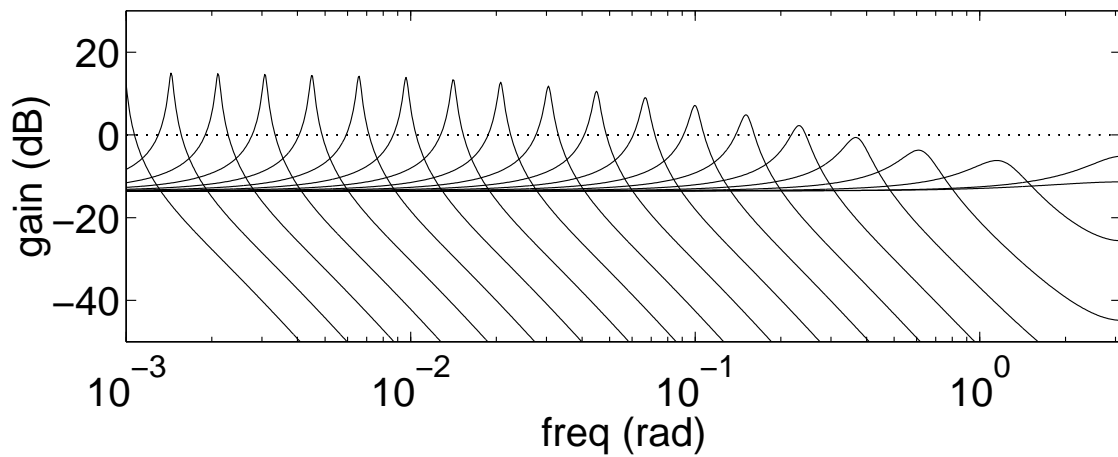This case used the separation table for stability reasons.

# Bode plots (medium Q)

## Bilinear transform with separation table:



## Backwards difference, no separation table:

# Notes

- Constant-Q is hard for digital filters

  – Constant-Q tracks are logarithmic spirals
  – These aren't "root-locus primitives"
  – The separation table does a good job making constant-Q (didn't necessarily expect this)

- Look at zero locations between the two cases (remembering that we would like to get rid of the table). First, review the parameters we have to control:

  – easiest: the zero locations
  – also easy: the relative open-loop pole locations (so far, have only really looked at all equal)
  – more expensive: frequency-to-pole and Q-to-feedback-gain mapping functions (or more esoteric mappings)
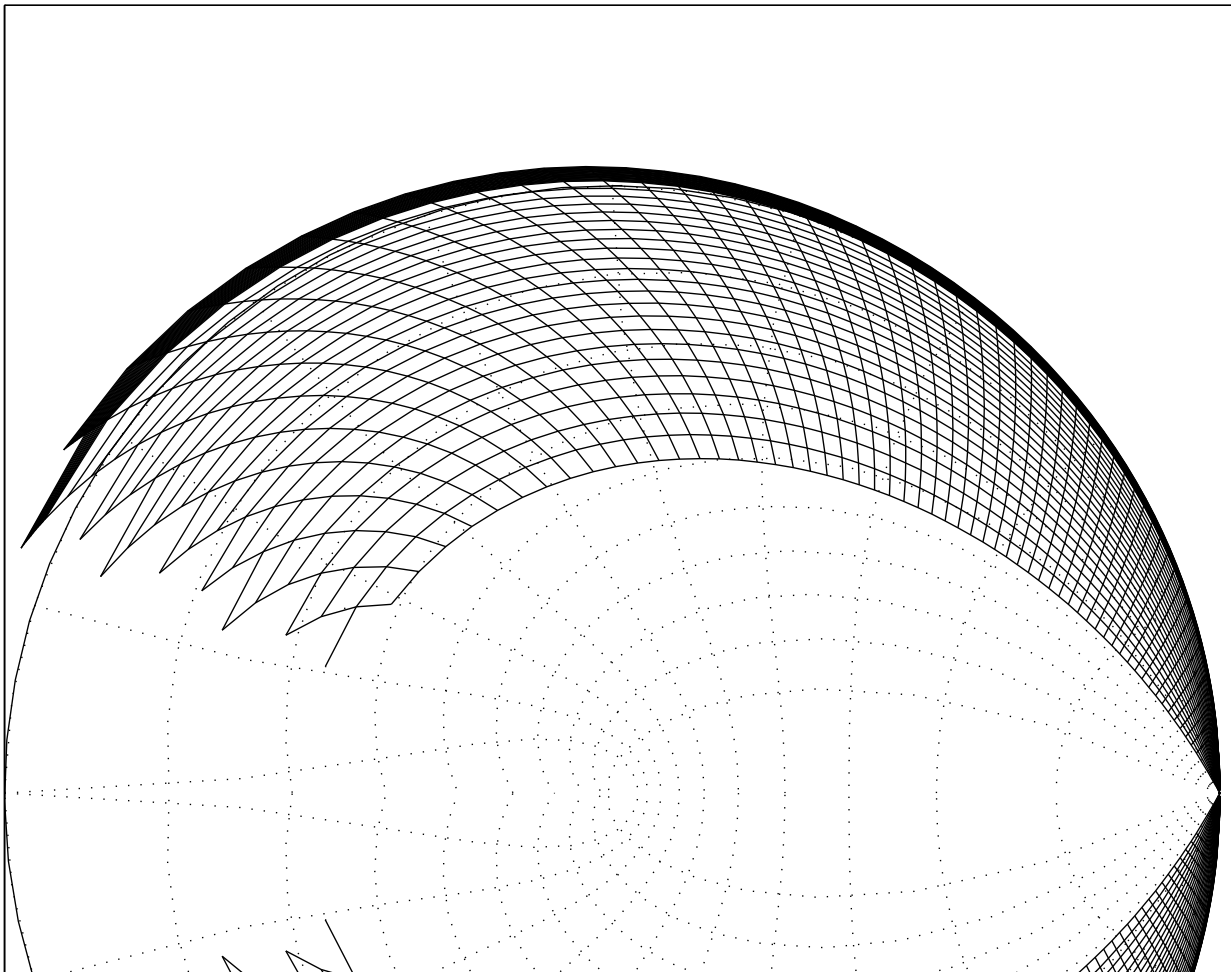
Note that

- backwards-difference yields zeros at $z = 0$

- bilinear transform yields zeros at $z = -1$
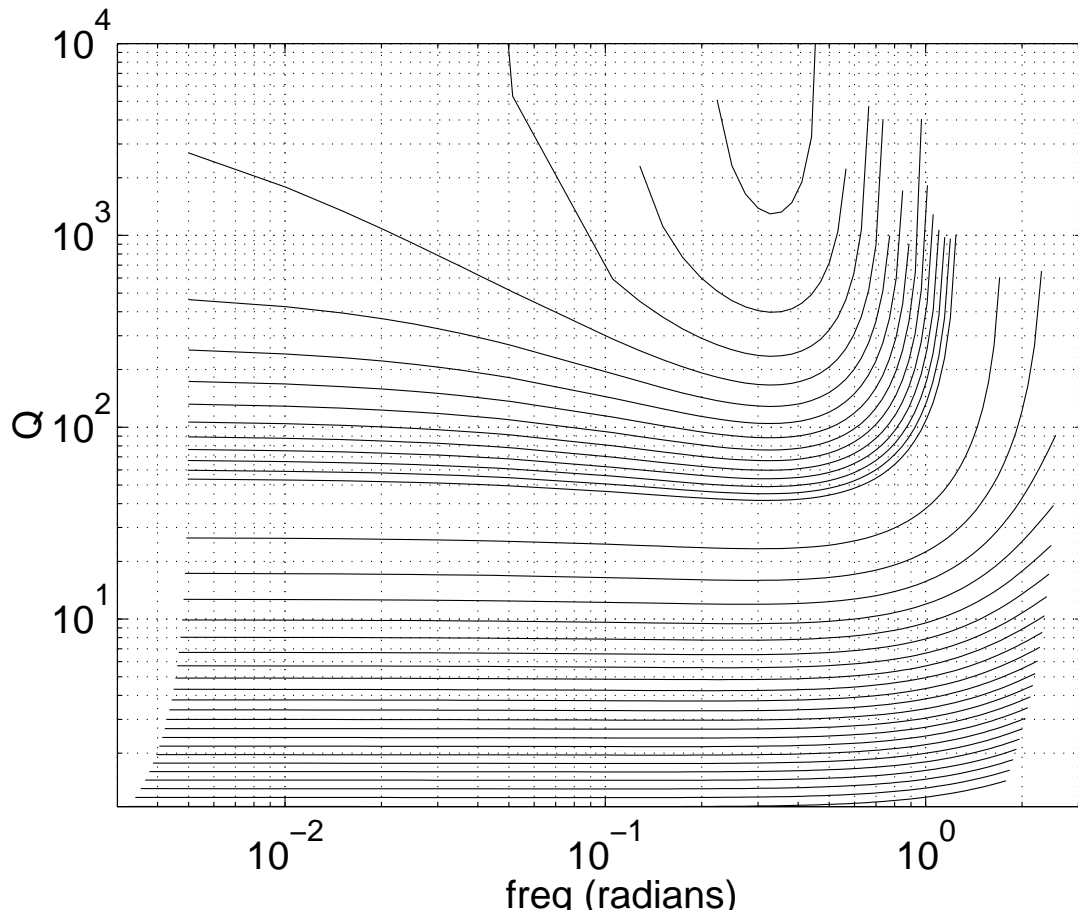
Try other locations...

# Discovery: There is a very good choice
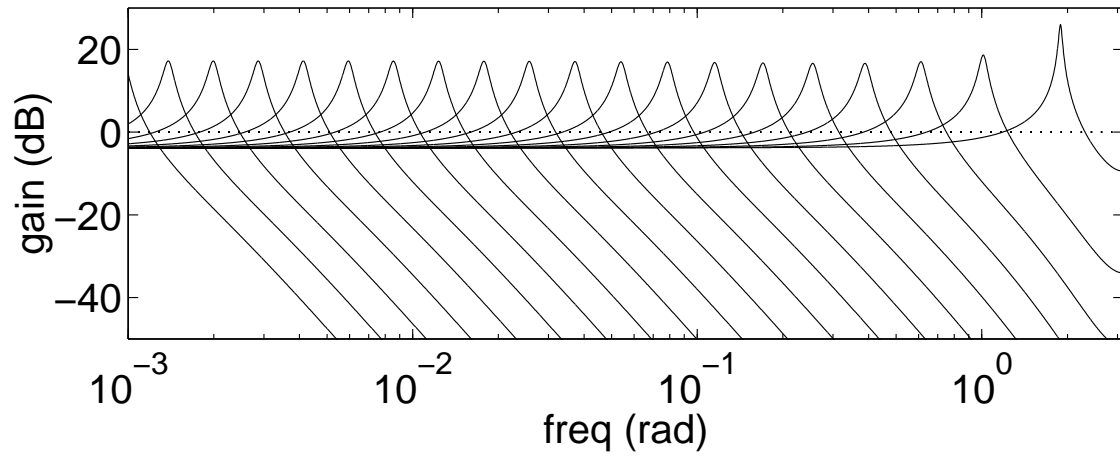
Zeros at $z = 0.3$
(Tim Stilson 1996)



Root Locus vs. Tuning

## **Zeros at $z = 0.3$ — Q vs. Tuning**



- Note that the Q-vs-frequency curves are pretty flat for $Q < 100$ for cut-off freq.s over most of the range

- This is quite good for not using a separation table

# Zeros at $z = 0.3$ — Bode Plot

# Farewell Bob Moog—-and Thank You!



Robert A. Moog (1934–2005)

# Online Resources

- This presentation:
  `http://ccrma.stanford.edu/~jos/Mohonk05/`

- Book chapter from which the proceedings paper was condensed:
  `http://ccrma.stanford.edu/~jos/pasp/-`
  `History_Enabling_Ideas.html`