|  |
| --- |
| > require(signal)  > # Compute and plot digital transfer functions  > library(signal)  > data(package="signal") # to see the data sets availale in the package signal  > ls("package:signal") # to list all of the objects in signal  [1] "an" "Arma" "as.Arma" "as.Zpg" "bartlett"  [6] "bilinear" "blackman" "boxcar" "butter" "buttord"  [11] "cheb1ord" "chebwin" "cheby1" "cheby2" "chirp"  [16] "conv" "decimate" "ellip" "ellipord" "fftfilt"  [21] "FftFilter" "filter" "FilterOfOrder" "filtfilt" "fir1"  [26] "fir2" "flattopwin" "freqs" "freqs\_plot" "freqz"  [31] "freqz\_plot" "gausswin" "grpdelay" "hamming" "hanning"  [36] "ifft" "impz" "interp" "interp1" "kaiser"  [41] "kaiserord" "levinson" "Ma" "medfilt1" "MedianFilter"  [46] "pchip" "poly" "polyval" "remez" "resample"  [51] "roots" "sftrans" "sgolay" "sgolayfilt" "specgram"  [56] "spencer" "spencerFilter" "triang" "unwrap" "Zpg"  [61] "zplane"  > lsf.str("package:signal") # to list all of the functions in signal  an : function (degrees)  Arma : function (b, a)  as.Arma : function (x, ...)  as.Zpg : function (x, ...)  bartlett : function (n)  bilinear : function (Sz, ...)  blackman : function (n)  boxcar : function (n)  butter : function (n, ...)  buttord : function (Wp, Ws, Rp, Rs)  cheb1ord : function (Wp, Ws, Rp, Rs)  chebwin : function (n, at)  cheby1 : function (n, ...)  cheby2 : function (n, ...)  chirp : function (t, f0 = 0, t1 = 1, f1 = 100, form = c("linear", "quadratic", "logarithmic"), phase = 0)  conv : function (x, y)  decimate : function (x, q, n = if (ftype == "iir") 8 else 30, ftype = "iir")  ellip : function (n, ...)  ellipord : function (Wp, Ws, Rp, Rs)  fftfilt : function (b, x, n = NULL)  FftFilter : function (b, n)  filter : function (filt, ...)  FilterOfOrder : function (n, Wc, type, ...)  filtfilt : function (filt, ...)  fir1 : function (n, w, type = c("low", "high", "stop", "pass", "DC-0", "DC-1"), window = hamming(n +  1), scale = TRUE)  fir2 : function (n, f, m, grid\_n = 512, ramp\_n = grid\_n/20, window = hamming(n + 1))  flattopwin : function (n, sym = c("symmetric", "periodic"))  freqs : function (filt, ...)  freqs\_plot : function (w, ...)  freqz : function (filt, ...)  freqz\_plot : function (w, ...)  gausswin : function (n, w = 2.5)  grpdelay : function (filt, ...)  hamming : function (n)  hanning : function (n)  ifft : function (x)  impz : function (filt, ...)  interp : function (x, q, n = 4, Wc = 0.5)  interp1 : function (x, y, xi, method = c("linear", "nearest", "pchip", "cubic", "spline"), extrap = NA,  ...)  kaiser : function (n, beta)  kaiserord : function (f, m, dev, Fs = 2)  levinson : function (x, p = NULL)  Ma : function (b)  medfilt1 : function (x, n = 3, ...)  MedianFilter : function (n = 3)  pchip : function (x, y, xi = NULL)  poly : function (x)  polyval : function (coef, z)  remez : function (n, f, a, w = rep(1, length(f)/2), ftype = c("bandpass", "differentiator", "hilbert"),  density = 16)  resample : function (x, p, q = 1, d = 5)  roots : function (x, method = c("polyroot", "eigen"))  sftrans : function (Sz, ...)  sgolay : function (p, n, m = 0, ts = 1)  sgolayfilt : function (x, p = 3, n = p + 3 - p%%2, m = 0, ts = 1)  specgram : function (x, n = min(256, length(x)), Fs = 2, window = hanning(n), overlap = ceiling(length(window)/2))  spencer : function (x)  spencerFilter : function ()  triang : function (n)  unwrap : function (a, tol = pi, dim = 1)  Zpg : function (zero, pole, gain)  zplane : function (filt, ...)  > # Filter H1(z)  > B1 <- c(0.2066,0.4131,0.2066) # Coefficients of numerator polynomial  > A1 <- c(1,-0.3695,0.1958) # Coefficients of denominator polynomial  > H1z <- freqz(B1,A1,100) # Compute the transfer function  >  > # Filter H2(z)  > B2 <- c(0.894,-1.789,0.894)  > A2 <- c(1,-1.788,0.799)  > H2z <- freqz(B2,A2,100)  >  > # Filter H3(z)  > B3 <- c(0.42,0,-0.42)  > A3 <- c(1,-0.443,0.159)  > H3z <- freqz(B3,A3,100)  >  > # Filter h4(z)  > B4 <- c(0.5972,0.4425,0.5972)  > A4 <- c(1,0.4425,0.1584)  > H4z <- freqz(B4,A4,100)  >  > # Convenience function to draw multiple plots  > hPlot <- function(H){  + text <- deparse(substitute(H)) # get the name of the filter for the title  + c <- substr(text,4,4)  + plot(H$f,abs(H$h),  + col="red",  + ylim=c(0,1),  + xlab="Normalized Frequency",  + ylab="Magnitude",  + main=paste("Filter H",c,"(z)",sep="")  + )  + }  >  > par(mfrow=c(2,2))  > plotList <- list(H1z,H2z,H3z,H4z)  > lapply(plotList,hPlot)  [[1]]  NULL  [[2]]  NULL  [[3]]  NULL  [[4]]  NULL  > #  > plot(H4z) # Look at the default plot |
|  |
| |  | | --- | |  | |