Comment to editor – I understand the motive to save space, but I strongly believe this reads much more easily with spaces between sections, at least until we're into Sponsor ballot. I have proposed spaces below, although another option is to revert to the spacing submitted with the original code. This is your option, however.
Tom

```
%%%%%%% MATLAB (R) script to compute TWDP %%%%%%%%%%%%%%%%%%%%%%
%% TP-2 test inputs
%% Units for all optical power values must match.

%% Transmit data file: The transmit data sequence is based on either of the TWDP test patterns defined in
%% Table 68-5. The file format is a single column of chronological ones and zeros with no headers or footers.
TxDataFile = 'path\datafilename';

%% Measured waveform. The waveform consists of exactly N samples per bit period T, where N is the
%% oversampling rate. The data sequence must be aligned with the waveform. The file format for the
%% measured waveform is a single column of chronological numerical samples, in optical power, with no headers
%% or footers.
MeasuredWaveformFile = 'path\waveformfilename';

%% OMA and steady-state ZERO power must also be specified.
MeasuredOMA          = [OMAvalue];           % Measured OMA, in optical power
SteadyZeroPower      = [ZEROvalue];               % Measured optical power, steady-state logic ZERO
OverSampleRate       = 16;                   % Oversampling rate

%% Simulated fiber response, modeled as a set of ideal delta functions with specified amplitudes in optical
%% power and delays in nanosecond
%% in two columns with no headers or footers. The number of test cases is determined by the TWDP requirements
%% in Table 68-3. The vector 'PCoefs' contains the amplitudes, and the vector 'Delays' contains the delays.
FiberResp       = load('fiber_case.txt');
PCoefs          = FiberResp(:, 2);
Delays          = FiberResp(:, 1);

%% Editor's note – These are static parameters that should not change once test is fully specified %%
SymbolPeriod    = 1/(10.3125);          % Symbol period (ns)
EFilterBW       = 7.5;                  % Front end filter bandwidth (GHz)
EqNf            = 100;                  % Number of feedforward equalizer taps
EqNb            = 50;                   % Number of feedback equalizer taps
EqDel           = ceil(EqNf/2);         % Equalizer delay
PAlloc          = 6.5;                  % Allocated dispersion penalty (dBo)
Q0              = 7.03;                 % BER = 10^(-12)

%% STEP 1 - Process waveform through simulated fiber channel %%%%%%%%%%%%%%%%%%
%% Load input waveforms
XmitData        = load(TxDataFile);
yout            = load(MeasuredWaveformFile);
PtrnLength      = length(XmitData);
TotLen          = PtrnLength*OverSampleRate;
Fgrid           = [-TotLen/2:TotLen/2-1].'/(PtrnLength*SymbolPeriod);
%% Process through fiber model. Fiber frequency response is normalized to 1 at DC
ExpArg          = -j*2*pi*Fgrid;
Hsys            = exp(ExpArg * Delays') * PCoefs;
Hx              = fftshift(Hsys/abs(Hsys(find(Fgrid==0))));
yout            = real(ifft(fft(yout).*Hx));

%% STEP 2 - Normalize OMA%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
yout    = (yout - SteadyZeroPower)/MeasuredOMA;

%% STEP 3 - Process signal through front-end antialiasing filter %%%%%%%%%%%%%%%%%%
%% Compute frequency response of front-end Butterworth filter
[b,a]   = butter(4, 2*pi*EFilterBW,'s');
H_r     = freqs(b,a,2*pi*Fgrid);
%% Process signal through front-end filter
yout    = real(ifft(fft(yout) .* fftshift(H_r)));

%% STEP 4 - Sample at rate 2/T %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
yout    = yout(1:OverSampleRate/2:end);

%% STEP 5 - Compute MMSE-DFE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% The MMSE-DFE filter coefficients computed below minimize mean-squared error at the slicer input.
%% The derivation follows from the fact that the slicer input over one period (which is the same as the
```

Deleted: (In the case used here, t
Deleted: derived from a PRBS9 data¶
%% sequence generated by polynomial x^9+x^4+1. (That is, the data sequence d(n) is given by¶
%% d(n)=d(n-9)+d(n-4), mod 2.) The sequence is initially aligned so that it starts with 9 ones. A zero¶
%% is inserted immediately after the string of eight zer ... [1]
Deleted: TxData.txt
Deleted: The current ... [2]
Deleted: ¶ ... [3]
Deleted:
Deleted:
Deleted: (For the P ... [4]
Deleted: ¶ ... [5]
Deleted: above has been
Deleted:
Deleted:)
Deleted: OMA and ste ... [6]
Deleted: G05.tx
Deleted: t
Deleted: % Measured ... [7]
Deleted: 3.8e-004
Deleted: 3.2e-004
Deleted: ¶ ... [8]
Deleted: ,
Deleted: .
Deleted:
Deleted: ¶ ... [9]
Deleted: FiberResp ... [10]
Deleted: S
Deleted: %%%%%%%%% ... [11]
Deleted:
Deleted: ¶
Inserted: ¶
Deleted: PRBSlength
Deleted: PRBSlength
Deleted: PRBSlength

```
%% period of the input data sequence) can be expressed as Z = (R+N)*W - X*[0 B]', where R and N are
%% Toeplitz matrices constructed from the signal and noise components, respectively, at the sampled out
%% put of the antialiasing filter, W is the feedforward filter, X is a Toeplitz matrix constructed from the
%% input data sequence, and B is the feedback filter. The optimal W and B minimize E[||Z-XIn||^2], where
%% XIn is the input data sequence, and the expectation operator refers to the Gaussian noise component of
%% Z. Compute the noise autocorrelation sequence at the output of the front-end filter and rate-2/T sampler.
%% Constuct a Toeplitz autocorrelation matrix.
N0      = SymbolPeriod/(2 * Q0^2 * 10^(2*PAlloc/10));
Snn     = N0/2 * fftshift(abs(H_r).^2) * 1/SymbolPeriod * OverSampleRate;
Rnn     = real(ifft(Snn));
Corr    = Rnn(1:OverSampleRate/2:end);
C       = toeplitz(Corr(1:EqNf));
%% Construct Toeplitz matrix from input data sequence
X       = toeplitz(XmitData, [XmitData(1); XmitData(end:-1:end-EqNb+1)]);
%% Construct Toeplitz matrix from signal at output of 2/T sampler.
%% This sequence gets wrapped by equalizer delay
R       = toeplitz(yout, [yout(1); yout(end:-1:end-EqNf+2)]);
R       = [R(EqDel+1:end,:); R(1:EqDel,:)];
R       = R(1:2:end, :);
%% Compute least-squares solution for filter coefficients
RINV    = inv(R'*R+PtrnLength*C);
P       = X'*(eye(PtrnLength) - R*RINV*R')*X;
P01     = P(1,2:EqNb+1);
P11     = P(2:EqNb+1,2:EqNb+1);
B       = -inv(P11)*P01';     % Feedback filter
W       = RINV*R'*X*[1;B];    % Feedforward filter
Z       = R*W - X*[0;B];      % Input to slicer

%% STEP 6 - Compute BER using semi-analytic method %%%%%%%%%%%%%%%%%%%%%%
MseGaussian    = W'*C*W;
Ber            = sum(0.5*erfc((abs(Z-0.5)/sqrt(MseGaussian))/sqrt(2)))/length(Z);

%% STEP 7 - Compute equivalent SNR %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% This function computes the inverse of the Gaussian error probability function. The built-in Matlab
%% function erfcinv() is not sensitive enough for the low probability of error case.
Q = inf;
if Ber>10^(-12)        Q = sqrt(2)*erfinv(1-2*Ber);
elseif Ber>10^(-300)   Q = 2.1143*(-1.0658-log10(Ber)).^0.5024;
end

%% STEP 8 - Compute penalty %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RefSNR = 10 * log10(Q0) + PAlloc;
fprintf(1,'TP2 penalty equals %5.4f dB\n', RefSNR-10*log10(Q));
```

Deleted: PRBSlength

Deleted: PRBSlength

Page 1: [1] Deleted	Tom Lindsay	24/02/2005 6:15 PM

```
derived from a PRBS9 data
%% sequence generated by polynomial x^9+x^4+1. (That is, the data sequence d(n) is given by
%% d(n)=d(n-9)+d(n-4), mod 2.) The sequence is initially aligned so that it starts with 9 ones. A zero
%% is inserted immediately after the string of eight zeros in the sequence. 2 bits are inverted at location 391
%% and 392. The resulting data sequence is inverted. Then the entire sequence is circularly right-shifted
%% 476 bits to align with the measured waveform specified below.)
```

Page 1: [2] Deleted	Tom Lindsay	24/02/2005 6:18 PM

```
The current waveform is for demonstration purposes only.
```

Page 1: [3] Deleted	Tom Lindsay	24/02/2005 6:27 PM

```
%%
```

Page 1: [4] Deleted	Tom Lindsay	24/02/2005 6:19 PM

```
 (For the PRBS9 case used
%% here, the oversampling rate is 16 and the waveform consists of 512*16 = 8192 samples.
```

Page 1: [5] Deleted	Tom Lindsay	24/02/2005 6:27 PM

```
%%
```

Page 1: [6] Deleted	Tom Lindsay	24/02/2005 6:21 PM

```
OMA and steady-state ZERO power
%% must also be specified.
```

Page 1: [7] Deleted	Tom Lindsay	24/02/2005 6:22 PM

```
 % Measured waveform samples, in optical power
```

Page 1: [8] Deleted	Tom Lindsay	24/02/2005 6:35 PM

```
%%
```

Page 1: [9] Deleted	Tom Lindsay	24/02/2005 6:30 PM

```
%%
```

Page 1: [10] Deleted	Tom Lindsay	24/02/2005 6:28 PM

```
FiberResp = [...
0.000000 0.65 0.88 0.51
0.072727 0.5 0.58 0.89
0.145455 0.91 0.89 0.29
0.218182 0.26 0.1 0.81];
```

Page 1: [11] Deleted	Tom Lindsay	24/02/2005 6:40 PM

%%%%%%%%%%%%%%%%%