Dual Dirac Jitter model

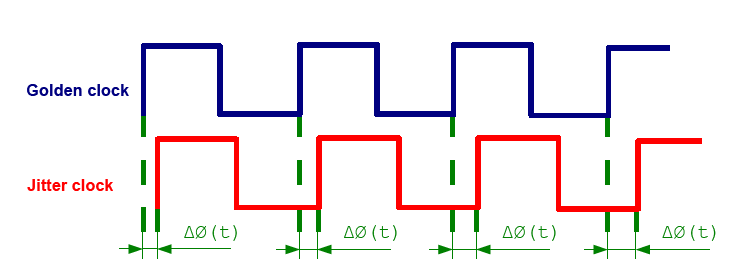
Niu Li

"Jun 17, 2016"

# Outline

* Jitter definition - short vs long term(TIE)
* Jitter category - RJ, DJ and details
* Review of statistics - PDF, CDF, erf, erfc
* BERT vs Scope
* Dual Dirac Model - vs
* Decompose method - Spectrum vs Tail Fit
* Scope Jitter setting

# TIE



* Deviation of the digital timing event from it is ideal position
* A specified number of observations.

# Jitter definition

* Period Jitter ()
  + Time difference between measured period and ideal period
* Cycle to Cycle Jitter ()
  + Time difference between two adjacent clock periods
  + Important for budgeting on-chip digital circuits cycle time
* Accumulated Jitter ()
  + Time difference between measured clock and ideal trigger clock
  + Jitter measurement most relative to high-speed link systems

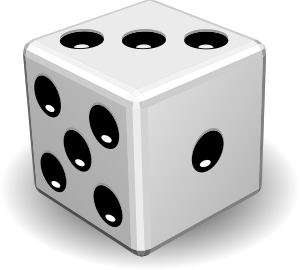
# Jitte Analysis Method

* Time domain
* Frequency domain
* Statistics domain
* Decompose to RJ and DJ

# Monte Carlo

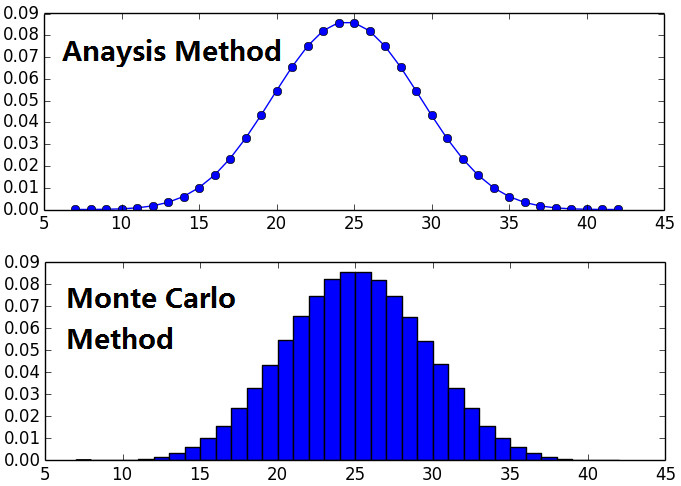
* Monte Carlo Simulation is a way of studying probability distributions with sampling
* Define a domain of possible inputs.
* Generate inputs randomly from a probability distribution over the domain.
* Perform a deterministic computation on the inputs.
* Aggregate the results.

# Dice gambling



* 7 dices , 6 faces.
* summarize the 7 dices points
* what is the PDF?

# Dice PDF



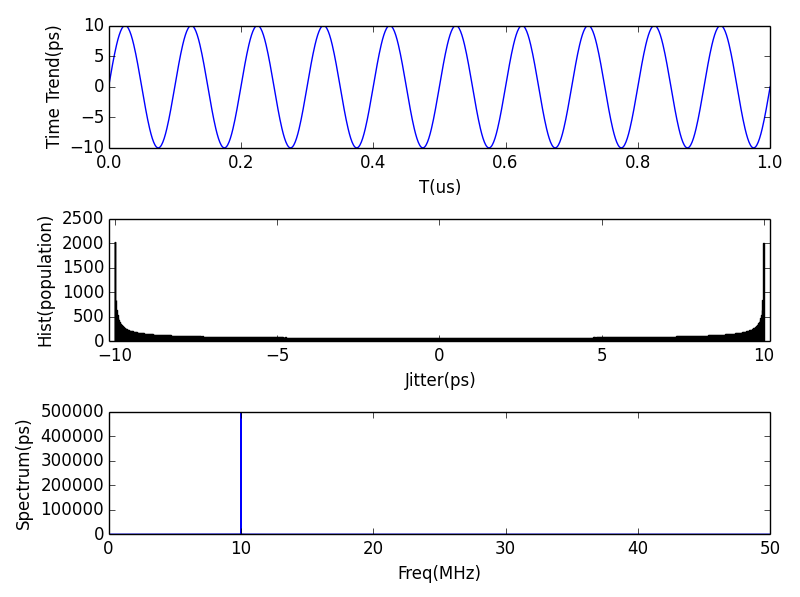
# Code

import numpy as np  
from numpy.random import random\_integers  
import matplotlib.pyplot as plt  
  
sample\_len = 1000000  
res = np.zeros(sample\_len)  
for i in range(sample\_len):  
 res[i] = np.sum(random\_integers(1,6,7))  
plt.hist(res,bins=35,normed=True)  
plt.show()

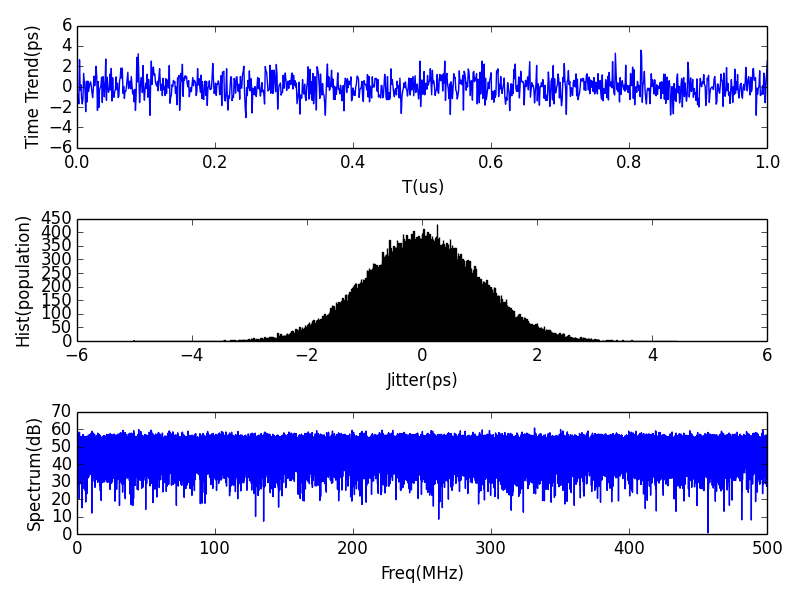
# Histogram(Simulation)

import numpy as np  
from numpy.fft import rfft, rfftfreq  
from numpy.random import normal  
import matplotlib.pyplot as plt  
  
PJ\_freq = 10e6 # 10MHz  
PJ\_amp = 10 # 10ps  
RJ\_rms = 1 # 1ps  
sample\_rate = 1e9 #1Gbps  
sample\_interval = 1./sample\_rate  
N\_cycle =1000  
pts = sample\_rate/PJ\_freq\*N\_cycle  
  
  
t = sample\_interval\*np.arange(pts)\*1E6  
  
# sin jitter  
plt.figure()  
tie\_sin = PJ\_amp\*np.sin(np.linspace(0,2\*np.pi\*N\_cycle,sample\_rate/PJ\_freq\*N\_cycle))  
plt.subplot(311)  
plt.plot(t,tie\_sin)  
plt.xlabel("T(us)")  
plt.ylabel("Time Trend(ps)")  
plt.xlim([0,1]) # only plot 1us  
plt.subplot(312)  
plt.hist(tie\_sin,bins=1000,normed=False)  
plt.xlabel("Jitter(ps)")  
plt.ylabel("Hist(population)")  
plt.xlim([-10.2,10.2])  
plt.subplot(313)  
plt.plot(rfftfreq(len(tie\_sin),1/sample\_rate)/1e6,np.abs(rfft(tie\_sin)))  
plt.xlabel("Freq(MHz)")  
plt.ylabel("Spectrum(ps)")  
plt.xlim([0,50])  
plt.tight\_layout()  
  
### random jitter  
plt.figure()  
tie\_normal = normal(loc=0,scale=RJ\_rms,size=pts)  
plt.subplot(311)  
plt.plot(t,tie\_normal)  
plt.xlabel("T(us)")  
plt.ylabel("Time Trend(ps)")  
plt.xlim([0,1]) # only plot 1us  
plt.subplot(312)  
plt.hist(tie\_normal,bins=1000,normed=False)  
plt.xlabel("Jitter(ps)")  
plt.ylabel("Hist(population)")  
plt.subplot(313)  
plt.plot(rfftfreq(len(tie\_normal),1/sample\_rate)/1e6,20\*np.log10(np.abs(rfft(tie\_normal))))  
plt.xlabel("Freq(MHz)")  
plt.ylabel("Spectrum(dB)")  
plt.tight\_layout()  
  
#Combined jitter  
plt.figure()  
tie\_combine = tie\_sin+tie\_normal  
plt.subplot(311)  
plt.plot(t,tie\_combine)  
plt.xlabel("T(us)")  
plt.ylabel("Time Trend(ps)")  
plt.xlim([0,1]) # only plot 1us  
plt.subplot(312)  
plt.hist(tie\_combine,bins=1000,normed=False)  
plt.xlabel("Jitter(ps)")  
plt.ylabel("Hist(population)")  
plt.subplot(313)  
plt.plot(rfftfreq(len(tie\_combine),1/sample\_rate)/1e6,20\*np.log10(np.abs(rfft(tie\_combine))))  
plt.xlabel("Freq(MHz)")  
plt.ylabel("Spectrum(dB)")  
plt.xlim([0,50])  
plt.tight\_layout()  
  
  
plt.show()

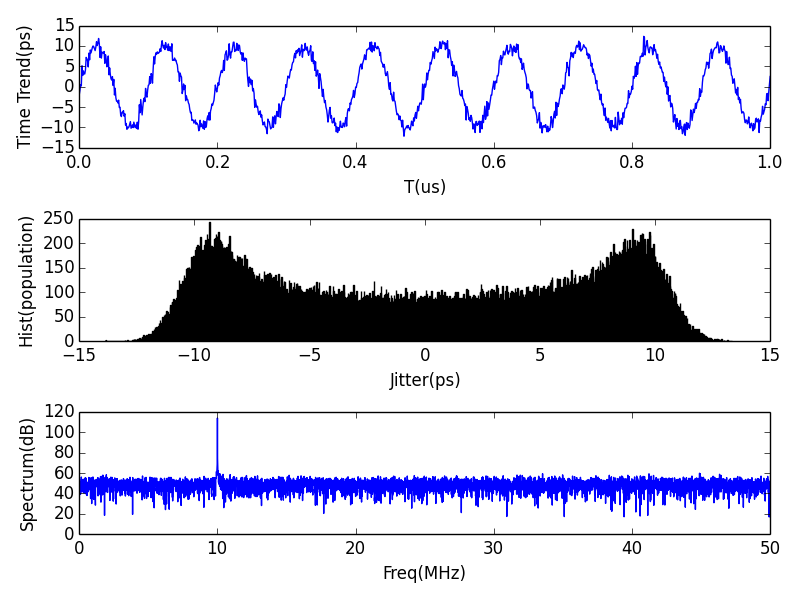
# Sinusoid



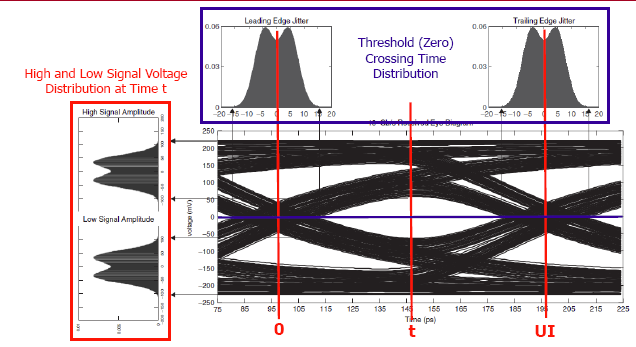
# Random(Gaussian)



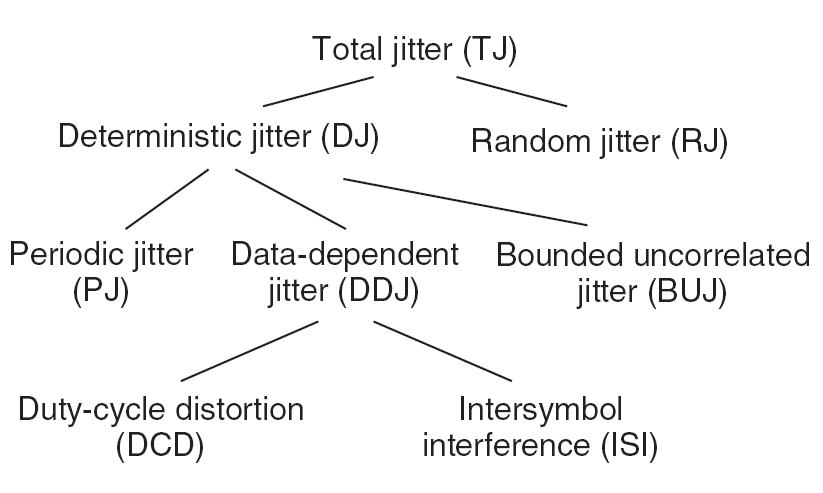
# Combined



# Jitter Histogram



# Jitter category



# Bounded vs Unbounded

* RJ is unbounded - long tail of Gaussian distribution
* Dj are bounded - must within a fixed range
* This is a significant differences, that's the reason why tail fit works

# Uncorrelated vs Dependent

* Uncorrelated or dependent is relative to data stream
* Uncorrelated jitter need "**convolve**" , dependent jitter need "**add**".
* RJ and DJ is uncorrelated
* DCD,ISI is data dependent
* PJ is also uncorrelated to data, but still "add"
* Crosstalk is BUJ(Bounded uncorrelated jitter)

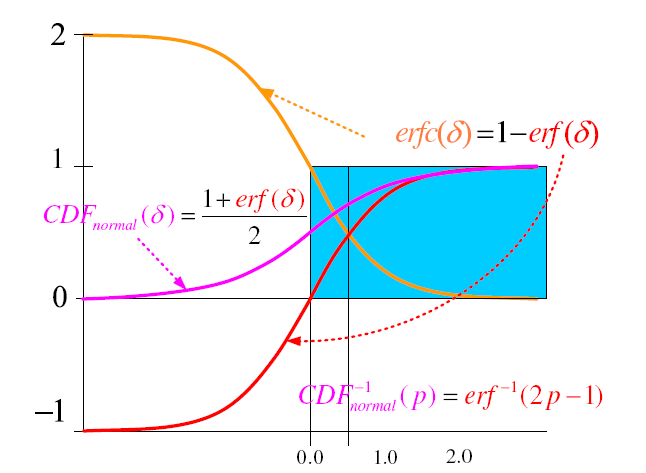
# Review of statistics

* PDF - The Probability density function, sometimes written .Histograms are closest in terms of a measurable probability density, and are said to approximate the PDF when normalized.
* CDF - The Cumulative Distribution Function is
* PDF of Gaussion.

# Statistics cont'd

* Error Function. . It is a basic a mathematical function closely related to the cumulative distribution function for a Gaussian.
* Complimentary Error Function. or
* Inverse Error Function, , This function has no closed analytic form but can be obtained from numerical methods.

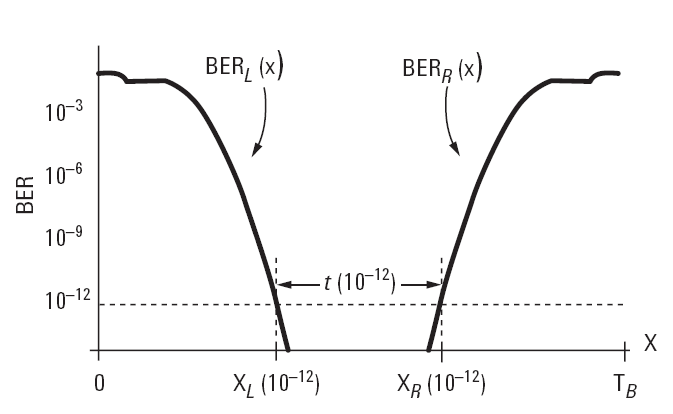
# ERF relations



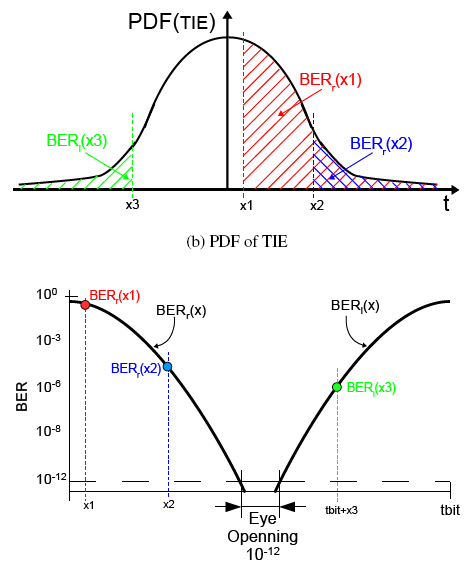
# BERT vs Scope

* BERT
  + Shift the sampling point across an entire bit time
  + Base on number of errors found
  + Accuratly measures the total jitter
* Scope
  + Get a limited number of samples
  + Good at determining the jitter components
  + Need post-process to estimate RJ

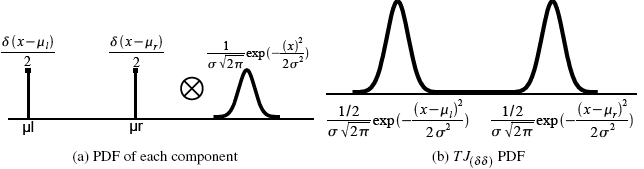
# Bathtub



# BER vs Jitter



# Dual Dirac Model



* An approximate method
* Quickly estimating of total jitter
* Not reflect the true jitter

# Assumption

1. Jitter can be separated into two categories, random jitter (RJ) and deterministic jitter (DJ).
2. RJ follows a Gaussian distribution and can be fully described in terms of a single relevant parameter, the rms() value of the RJ distribution.
3. DJ follows a finite, bounded distribution.
4. DJ follows a distribution formed by two Dirac-delta functions.
5. Jitter is a stationary phenomenon. Give the same result regardless of when that time interval is initiated.

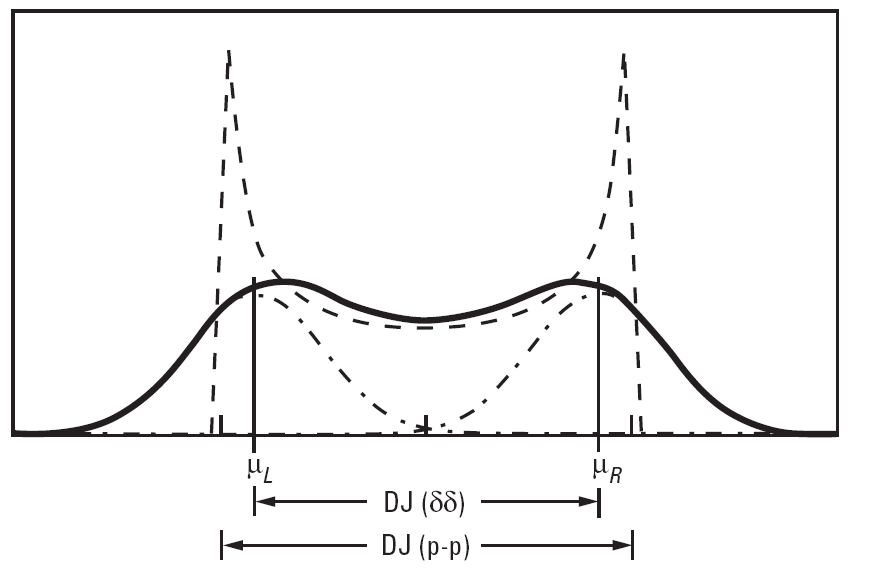
# Math Expression

* PJ pdf is
* RJ pdf is
* TJ combined by convolve, as it is uncorrelated
* RJ RMS combine:
* DJ pp combine:

# vs

* Dual-Dirac DJ is a completely different quantity than the peak-to-peak DJ(very confused)
* To distinguish the two use the notation as the real jitter, as dual-Dirac model DJ
* never follows the simple dual-Dirac distribution
* is a model dependent quantity that must be derived under the assumption that DJ follows a distribution formed by two Dirac-delta. Generally,

# vs cont'd



# TJ vs BER

if BER if 1e-12, we could get the factor as 14.07 by the following code. Most of time we use a look-up table

from scipy.special import erfcinv  
import numpy as np  
BER = 1E-12  
print 2\*np.sqrt(2)\*erfcinv(2\*BER)

# Jitter decompose

* Spectral Method
* Tail Fit method

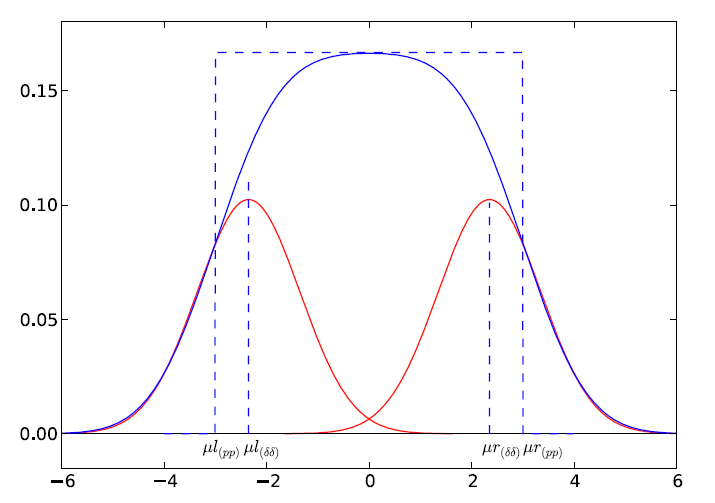
# Spectral Method

* FFT of TIE(t)
* spectral “peaks” are manifestations of deterministic jitter, and thus provide a measure of DJ
* The remaining spectrum, or “noise floor” accounts for all of RJ
* Remove the "peaks", and IFFT the "noise floor" get RJ
* Deconvolve and get DJ

# Spectral problem

* BUJ like crosstalk is will impact noise floor
* Non-Stationary Periodic aggressors (like Spread Spectrum) manifest broadly in the jitter spectrum and cannot always be identified as “peaks”
* In effect, any wide-band aggressor which contributes bounded timing fluctuations will be indistinguishable from the noise floor, and consequently “counted” as RJ.
* Then it is easy to overestimations of Tj

# Tail fit method



# Tail fit difficulties

* Exponential waveform fit. - Q-scale
* Find the region of pure Gaussian PDF. - Re-normalized Q-scale
* Need enough samples to get more long tail data.

# Q-scale

* A smart method transform the exponential waveform to linear
* Q-scale is a simple coordinates transformation. Similar method like Cartesian coordinates to Polar coordinates, time domain to frequency domain. Different view point of the same physical object.
* Replace x with in BER expression, we get Q scale.

# Math deduction

At the region far from DJ(long tail), the distribution is Gaussion, it also have left and right part. Just take left side as example.

Let , we get

# Math cont'd 1

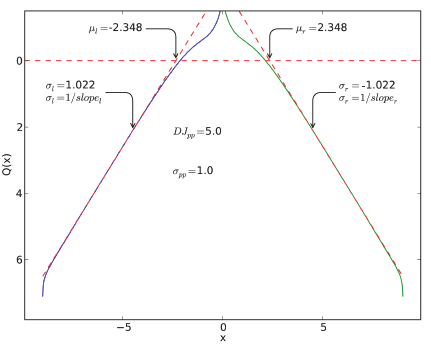
Remember that complementary error function is given by

then we could rewrite as

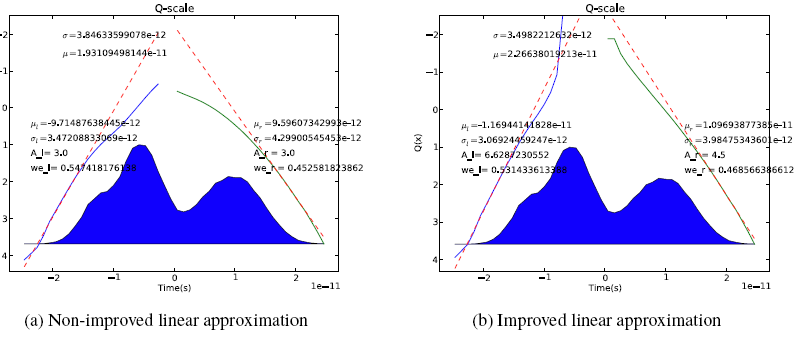
inverse the function, we get

# Math cont'd 2

compare the 2 equation, the target is fit the 2.



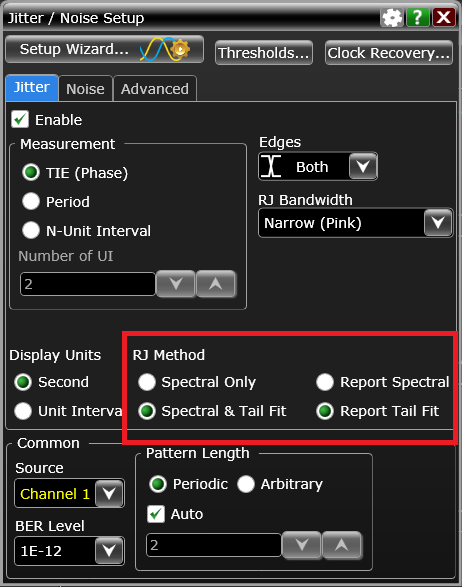
# Q-scale fit



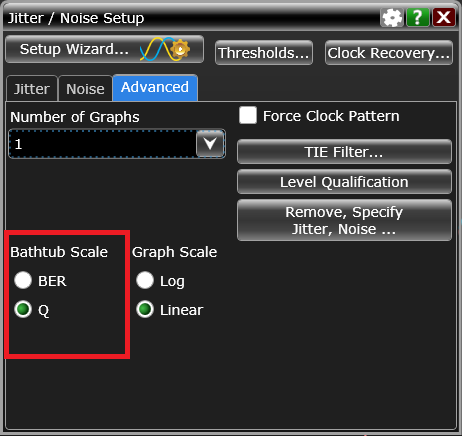
# Pros of tail fit

* The noise like or flat spectrum jitter source like crosstalk and SSC could be remove from RJ
* If a system will too much crosstalk, it is better use tail fit method, otherwise spectrum method could be used.

# Scope setting



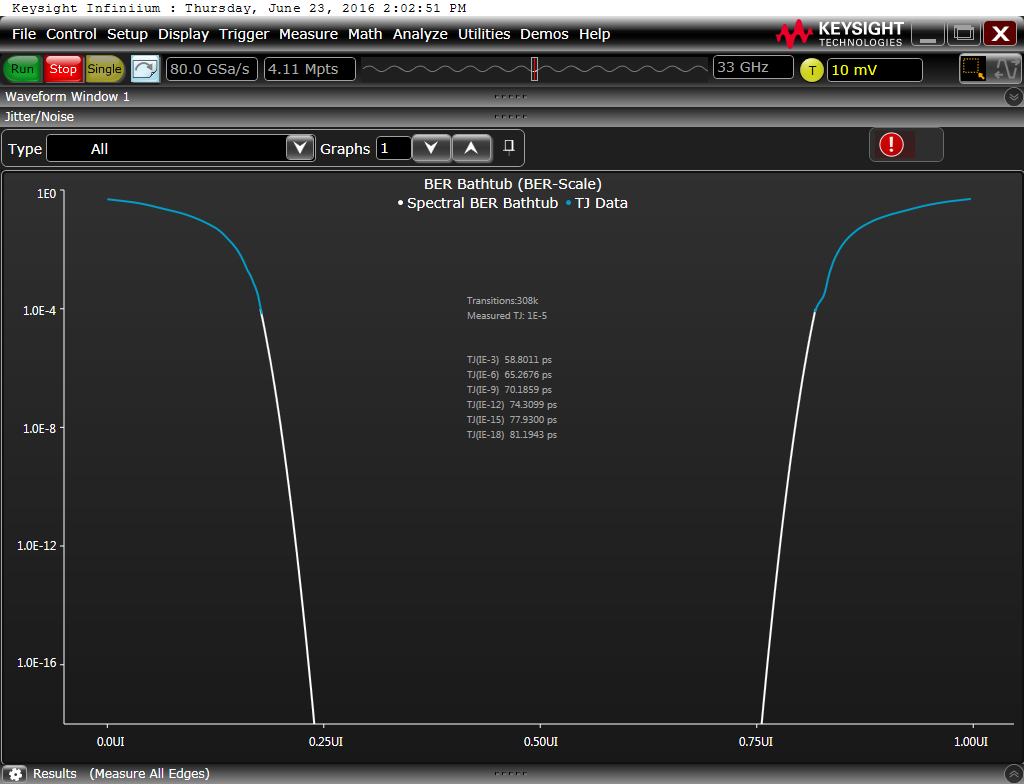
# Q-Scale option



# Split threshold

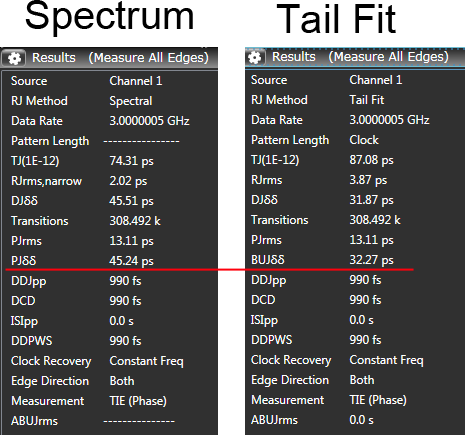


# BER scale



# Q scale

# Jitter Result



# Summary

* Dual-Dirac model is for estimate RJ, aks. extrapolation.
* RJ is the easy split than DJ, as its PDF feature.
* Both spectrum and tail fit have pros and cons