Thermal Noise Analysis Script (TNAS) v0.1 User's Manual (draft)

Nicholas A. Masluk
namasluk@us.ibm.com

January 6, 2022

Contents

1 Theory

1.1 Johnson-Nyquist Noise

Thermal noise in electrical dissipation was first described experimentally by Johnson [1] and theoretically by Nyquist [2] in 1928. For an arbitrary electrical circuit with two nodes, at uniform temperature T, with impedance $Z(f)$ (admittance $Y(f) = 1/Z(f)$), the one-sided noise-voltage power spectral density (PSD) across the nodes at frequency f is given by

$$S_{vv}(f,T) = 4\Re[Z(f)]\epsilon(f,T), \tag{1}$$

and the noise-current PSD through a short across the nodes is given by

$$S_{ii}(f,T) = 4\Re[Y(f)]\epsilon(f,T), \tag{2}$$

where $\epsilon(f,T)$ is the average energy per degree of freedom (discussed below). A noise-voltage source of $\sqrt{S_{vv}(f,T)}$ in series with $Z(f)$, and a noise-current source of $\sqrt{S_{ii}(f,T)}$ in parallel with $Y(f)$ are, respectively, the Thevanin (Figure 1) and Norton (Figure 2) representations of the thermal noise[1]. Both representations are equivalent and may be used interchangeably.

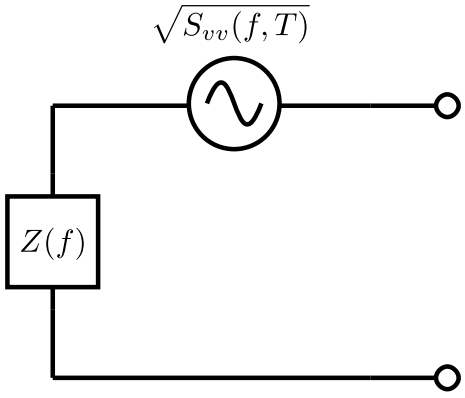

Figure 1: Thevanin representation of impedance $Z(f)$ with integral thermal noise source.

[1] In SI units, the voltage source $\sqrt{S_{vv}(f,T)}$ has units of $\mathrm{V}/\sqrt{\mathrm{Hz}}$, and the current source $\sqrt{S_{ii}(f,T)}$ has units of $\mathrm{A}/\sqrt{\mathrm{Hz}}$. While these units may appear strange at first,

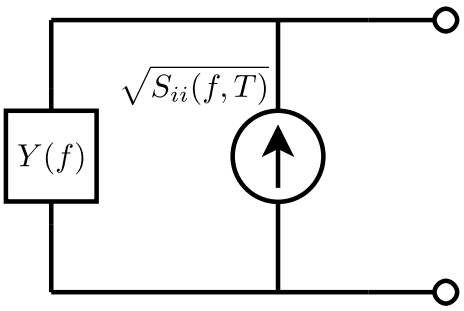

Figure 2: Norton representation of admittance $Y(f)$ with integral thermal noise source.

The average energy per degree of freedom is given by

$$\epsilon(f,T) = \frac{hf}{e^{hf/k_BT} - 1}, \tag{3}$$

where k_B is Boltzmann's constant and h is Planck's constant. When $hf \ll k_BT$, which is true for most conventional electronics, we are in the low-frequency / high-temperature limit where $\epsilon(f,T) \simeq k_BT$, and noise is white for pure resistors. In this regime, for a noisy resistor R we arrive at the classical expressions for Johnson-Nyquist white noise

$$\sqrt{S_{vv}(f,T)} \simeq \sqrt{4k_BTR}, \tag{4}$$

$$\sqrt{S_{ii}(f,T)} \simeq \sqrt{\frac{4k_BT}{R}}. \tag{5}$$

consider that when these quantities are across or through a load, the power spectral density delivered (PSD) to that load is given by squaring the noise-voltage and dividing by the load resistance, or squaring the noise-current and multiplying by the load resistance. The SI units of this PSD are W/Hz. To get the total power (in SI units of W), the PSD must be integrated across the frequency bandwidth coupling the noise source and load.

Note that $S_{vv}(f,T)$ and $S_{ii}(f,T)$ are both proportional to power by a constant scale factor. Because of this proportionality, both $S_{vv}(f,T)$ and $S_{ii}(f,T)$ are referred to as power spectral densities, but require multiplication or division by a resistance in order to properly convert into a power per unit frequency. This nomenclature is confusing, but in this text we refer to $S_{vv}(f,T)$ as the noise-voltage PSD, $S_{ii}(f,T)$ as the noise-current PSD, $\sqrt{S_{vv}(f,T)}$ as the noise-voltage, $\sqrt{S_{ii}(f,T)}$ as the noise-current, and actual powers per unit frequency simply as the PSD.

At microwave frequencies and cryogenic temperatures, Johnson-Nyquist noise no longer appears white in pure resistors and begins roll off with increasing frequency. This roll-off is shown in plotting $\epsilon(f,T)/k_BT$ in Figure 3. Were it not for this roll off, which comes about due to energy quantization, the total noise power would be infinite; this is the circuit equivalent of the early 20$^{\text{th}}$ century ultraviolet catastrophe in predicting black-body radiation, and resulted in the start of quantum theory.

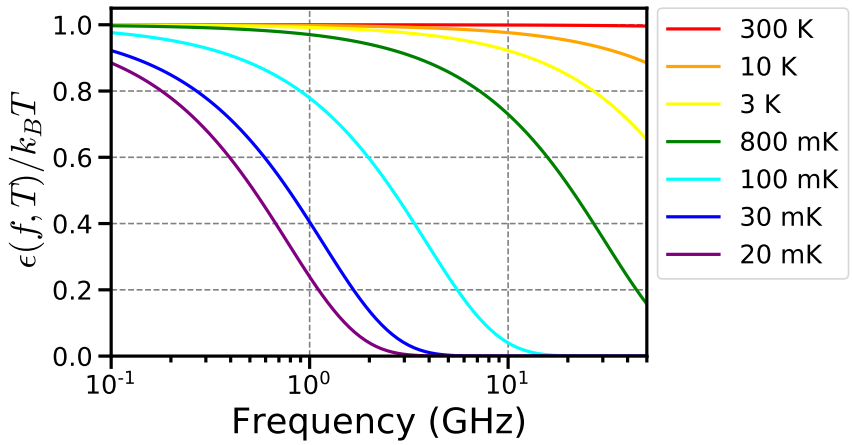

Figure 3: Roll-off of the average energy per degree of freedom can be seen as hf approaches and exceeds k_BT.

Electrical noise generated in resistors is the one-dimensional equivalent of blackbody radiation; the underlying mechanism of their generation is one and the same. The link between Johnson-Nyquist noise and blackbody radiation was described by Dicke [3], where the difference in frequency dependence can be attributed to the change in radiation pattern of a matched antenna enclosed in a blackbody cavity. At low frequency, Johnson-Nyquist noise of a resistor is white (independent of f), and blackbody radiation is proportional to f^2, while the solid angle of an antenna's radiation pattern is proportional to $1/f^2$. A more fundamental description of the source of thermal noise is given by the fluctuation-dissipation theorem, which is out of the scope of this overview.

The Thevanin and Norton equivalent representations of Figure 1 and Figure 2 describe the thermal noise of any arbitrary circuit with two nodes.

Assuming the noise sources are uncorrelated, each noise source may be analyzed independently.

For example, take three resistors R_1, R_2, and R_3 placed in a series loop. The total noise-voltage PSD across resistor R_1 is given by summing the individual noise power contributions from each resistor. Where resistor R_α generates noise-voltage PSD $S_{vv,\alpha}(f, T_\alpha)$, this goes through a voltage division $\frac{R_1}{R_1+R_2+R_3}$ to generate noise-voltage PSD $S_{vv,\alpha}(f, T_\alpha)\left(\frac{R_1}{R_1+R_2+R_3}\right)^2$ across R_1. When evaluating the noise across R_1 from its own noise source, we must keep in mind that the noise-voltage source is integral to the resistor, so we consider the voltage across the full noisy resistor model of Figure 1, not just the resistive element. Therefore, the noise-voltage PSD across R_1 from R_1's noise-voltage source is $S_{vv,1}(f, T_1)\left(1 - \frac{R_1}{R_1+R_2+R_3}\right)^2$. The total noise-voltage PSD across R_1 is $S_{vv,1}^{\text{Total}}(f) = S_{vv,1}(f, T_1)\left(1 - \frac{R_1}{R_1+R_2+R_3}\right)^2 + \sum_{\alpha=2}^{3} S_{vv,\alpha}(f, T_\alpha)\left(\frac{R_1}{R_1+R_2+R_3}\right)^2$.

The noise PSD flow from an arbitrary aggressor circuit with impedance $Z_\alpha(f)$ and noise-current PSD $S_{ii,\alpha}(f)$ connected to a victim circuit with impedance $Z_\beta(f)$ is given by

$$P_{\alpha\to\beta}(f) = S_{ii,\alpha}(f)\left|\left(\frac{Z_\alpha(f)Z_\beta(f)}{Z_\alpha(f)+Z_\beta(f)}\right)^2\right|\Re[Z_\beta^{-1}(f)]. \tag{6}$$

The power flow from aggressor to victim and victim to aggressor is equal at frequency f if the victim is at uniform temperature

$$T_{\text{equ},\beta}(f) = \frac{hf}{k_B \ln\left(\frac{4hf}{S_{ii,\alpha'}(f)|Z_\beta(f)|}+1\right)}, \tag{7}$$

where

$$S_{ii,\alpha'}(f) = \frac{S_{ii,\alpha}(f)Z_\alpha(f)}{\left|\left(\frac{Z_\alpha(f)Z_\beta(f)}{Z_\alpha(f)+Z_\beta(f)}\right)^2\right|\Re[Z_\alpha^{-1}(f)]}. \tag{8}$$

The function $T_{\text{equ},\beta}(f)$ gives the thermal equilibrium temperature at the narrow frequency space about f. In general, $T_{\text{equ},\beta}(f)$ will have a frequency dependence, unless all dissipitive elements within the aggressor circuit are at a uniform temperature. Absolute thermal equilibrium of a victim circuit at uniform temperature is given when total power flow is equal between aggressor and victim. If the aggressor circuit is at uniform temperature T_α,

then $T_{\text{equ},\beta}(f) = T_\alpha$ for all f; in this case, the color temperature of the noise from the aggressor circuit is pure (there is a continuous range of "primary colors", given by every possible temperature). When the aggressor circuit is not at a uniform temperature, $T_{\text{equ},\beta}(f)$ will be frequency dependent, and a victim circuit in thermal equilibrium with the aggressor circuit will receive more noise power than it sends in some frequency bands, and send more noise power than it receives in others, but total noise power flow is balanced. In the case of circuits not at uniform temperature, their noise color temperature is mixed.

TODO

Noise temperature definition and its shortcomings, difference between noise temperature and equilibrium temperature; noise temperature only equals equilibrium temperature when impedance of aggressor circuit is strictly real.

Demo where noise temperature is high but equilibrium temp is low and vice versa. Can't trust noise temperature by itself, passive attenuators do not thermalize even though they drop the noise temperature.

The steady-state temperature of a resonator (qubit) is equal to the equilibrium temperature of the effective resistance that sets the total quality factor of the resonator (T_1 of the qubit).

distributed...

1.2 Noise Analysis Script

In the noise analysis script, the netlist of a circuit is imported, and a victim component is selected by the user. The netlist is moified so that voltage sources are replaced with shorts, and current sources are replaced with opens. Following this, a current source is placed in parallel with each resistive element one at a time (the aggressors), and the voltage response across the victim to the aggressor current source is determined (a transimpedance) through a SPICE simulation. The simulation calculates this response for each frequency of interest. In total, the number of SPICE simulations executed equals the number of resistive components in the circuit (which includes the victim's response to itself), with each simulation stepping through frequencies of interest.

With all of the responses (transimpedances) calculated, temperatures are provided for all resistive components (including the victim, which itself is also an aggressor if it is resistive), which determines the noise-current PSD they will produce. Given this and the transimpedances calculated by SPICE

(in volts across victim per amp from noise-current source), all of the noise-voltages produced across the victim from each of the aggressors are summed in quadrature (the powers are added, rather than the voltages; these noise sources are considered to be uncorrelated). This gives the total noise-voltage PSD produced across the victim due to all thermal noise in the circuit.

2 Getting Started

The noise analysis script is written in Python, and comes as a Jupyter notebook. The script may be run directly in Python (without Jupyter) with only minor modification after copying and pasting two cells to a text file (removal of a few non-ASCII characters, and re-formatting of the plots). The script is compatible with both Python 2 and Python 3, and has been tested with 64-bit Python 2.7 and Python 3.7, but will likely work with much earlier versions. The required Python packages are (all included with the Anaconda Python Distribution): `math`, `numpy`, `scipy`, `sys`, `os`, `tempfile`, `shutil`, `subprocess`, `matplotlib`.

In addition to Python, a SPICE platform is required. Both ngspice and WRspice have been tested and work equivalently well. Other SPICE packages may work, but have not been tested.

While netlists can be written manually, it is much more convenient to use schematic capture software to draw the system. The following software will generate netlists compatible with the analysis script:

- **Keysight Advanced Design System (ADS)**
  Windows and Linux
  License subscription required
  https://www.keysight.com/us/en/products/software/pathwave-design-software/pathwave-advanced-design-system.html

- **Eeschema**
  Windows, Linux, and MacOS
  Freeware
  https://www.kicad-pcb.org/discover/eeschema/

- **LTspice**
  Windows and MacOS
  Freeware

https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html

- **Qucs**
  Windows, Linux, and MacOS
  Freeware
  http://qucs.sourceforge.net/

- **QucsStudio**
  Windows
  Freeware
  http://dd6um.darc.de/QucsStudio/qucsstudio.html

2.1 Installation of Required Components

2.1.1 Windows

- Install Python. The Anaconda Python 3 distribution is the most commonly used Python platform by IBM Quantum. The Anaconda Python Distribution may be downloaded at
  https://www.anaconda.com/distribution/

- Install SPICE. The analysis script has been tested with ngspice (http://ngspice.sourceforge.net/) and WRspice (available in the XicTools package at http://wrcad.com/xictools/index.html; you must install both xictools_wrspice and gtk2-bundle). Other SPICE platforms may work, but have not been tested.

2.1.2 Linux

- Install Python. The Anaconda Python 3 distribution is the most commonly used Python platform by IBM Quantum. The Anaconda Python Distribution may be downloaded at
  https://www.anaconda.com/distribution/

- Install SPICE. The analysis script has been tested with ngspice (http://ngspice.sourceforge.net/) and WRspice (available in the XicTools package at http://wrcad.com/xictools/index.html). Other SPICE platforms may work, but have not been tested.

2.1.3 Mac OS X

Note The analysis script has not been tested on Mac OS X. However, all of the same components tested in both Windows and Linux are available for Mac, so it is expected to work.

- Install Python. The Anaconda Python 3 distribution is the most commonly used Python platform by IBM Quantum. The Anaconda Python Distribution may be downloaded at
  https://www.anaconda.com/distribution/
- Install SPICE. The analysis script has been tested with ngspice (http://ngspice.sourceforge.net/) and WRspice (available in the XicTools package at http://wrcad.com/xictools/index.html). Other SPICE platforms may work, but have not been tested.

2.2 Defining the SPICE command

2.2.1 Windows

Open the noise analysis notebook in Jupyter. The first code cell contains the core of the analysis script, and should not be edited. In the second code cell, under the "User input" header, edit the `spice_cmd =` line to equal the command used for your chosen SPICE platform. In Windows the SPICE command is unlikely to be in the search path, so the full path should be given. Using the default installation path for WRspice we have
`spice_cmd = r'c:\usr\local\xictools\wrspice.current\bin\wrspice.bat'`.
The Windows binaries for ngspice are distributed as a ZIP file that can be extracted anywhere, and we use the console executable `ngspice_con.exe` (`ngspice.exe` will work, but it will launch a GUI window, greatly slowing down execution speed). If we extract directly to `c:\` our line will be
`spice_cmd = r'c:\Spice64\bin\ngspice_con.exe'`.
The "`r`" preceding the string literal gives us a "raw string", allowing the backslashes to be interpreted as backslashes, rather than as escape characters.

2.2.2 Linux and Mac OS X

Open the noise analysis notebook in Jupyter. The first code cell contains the core of the analysis script, and should not be edited. In the second code

cell, under the "User input" header, edit the `spice_cmd =` line to equal the command used for your chosen SPICE platform. Simply entering
`spice_cmd = 'wrspice'`
for WRspice or
`spice_cmd = 'ngspice'`
for ngspice will work as long as the executable is in the search path. If not in the search path, the full path must be given.

3 Usage

The first code cell of the script notebook contains all of the core code, and must be executed first. After this first execution, all definitions are loaded in memory, so this cell does not need to be executed again. Under ordinary use, this cell never needs to be edited.

The second code cell contains all of the inputs required by the user to run a simulation.

3.1 Generating Netlists

After drawing a schematic of the system to simulate, follow these steps to generate a netlist usable by the noise analysis script.

3.1.1 Keysight Advanced Design System (ADS)

- "Simulate" menu ⇒ "Generate Netlist"
- Save to a file

3.1.2 Eeschema

- "Tools" menu ⇒ "Generate Netlist File"
- "Spice" tab
- Un-check "Use net number as net name"
- Click "Generate" button to save a SPICE netlist

3.1.3 LTspice

- Add a ".op" SPICE directive
- Under "Simulate" menu click "Run" (or click the "Run" icon)
- In the directory containing the LTspice schematic (.asc file) a .net file with a matching filename prefix will be generated. Copy this file to another location. Closing LTspice will delete this file.

3.1.4 Qucs

- "Simulation" menu ⇒ "Simulate" (or press the gear icon). This is required to generate a netlist, but no simulation needs to be set up.
- "Simulation" menu ⇒ "Show Last Netlist"
- Go to "File" ⇒ "Save as..." to save the netlist to a text file (or copy and paste the entire contents to a text file)

The header "# Qucs " must be included at the top in order for the analysis program to recognize this netlist as originating from Qucs.

3.1.5 QucsStudio

- "Simulation" menu ⇒ "Simulate" (or press the gear icon). This is required to generate a netlist, but no simulation needs to be set up.
- "Simulation" menu ⇒ "Show Last Netlist"
- Go to "File" ⇒ "Save as..." to save the netlist to a text file (or copy and paste the entire contents to a text file)

The header "# QucsStudio " must be included at the top in order for the analysis program to recognize this netlist as originating from QucsStudio.

3.2 User Inputs

`spice_cmd` is a string for the command to be used

`fname_netlist` is a string giving the filename and path to the netlist file to be analyzed.

`refdes_victim`
`default_temperature`
`temperature_dict`
`aggressor_groups`
`group_temperature_dict`
`freq_start`
`freq_stop`
`points_per_dec`
`show_results` is a Boolean value

3.3 Program Outputs

The primary outputs from the analysis script is produced by `do_analysis()`.

3.3.1 Raw Outputs

`netlist`
`noise_func_dict`
`temperature_dict`
`response_dict`
`svv_total`
`noise_ratio_dict`
`group_noise_ratio_dict`
`power_abs`
`power_abs_err`
`noise_sys_to_victim`
`noise_victim_to_sys`
`noise_equ_victim_to_sys`
`T_equ_i`
`T_noise`
`T_noise_equ`
`T_equ`
`svv_total_equ`

3.3.2 Plots

3.4 Examples

4 Script Structure

This section describes the details of the analysis script. This section is not necessary to read in order to use and understand results of the script, but it serves as documentation in order to fix bugs or further develop the script.

Upon execution of the analysis script, input parameters from the user (described in subsection 3.2) are passed to `do_analysis()`. The `do_analysis()` function executes all of the major functions of the script in the appropriate order, then ultimately returns the collection of results in a tuple. The results returned by `do_analysis()` are described in subsubsection 3.3.1.

This collection of results is then optionally (if the user defined Boolean `show_results` is true) passed on to `display_results()`.

References

[1] J. B. Johnson, "Thermal agitation of electricity in conductors," *Phys. Rev.*, vol. 32, pp. 97–109, Jul 1928.

[2] H. Nyquist, "Thermal agitation of electric charge in conductors," *Phys. Rev.*, vol. 32, pp. 110–113, Jul 1928.

[3] R. H. Dicke, "The measurement of thermal radiation at microwave frequencies," *Review of Scientific Instruments*, vol. 17, no. 7, pp. 268–275, 1946.