SAND2020-11306C

Using `pygsti` for quantum processor characterization and benchmarking

Sandia National Laboratories | QUANTUM PERFORMANCE LAB

QPL

Erik Nielsen, *Timothy Proctor, Kenneth Rudinger, Kevin Young, and Robin Blume-Kohout*

First International Workshop on Quantum Computing Software in conjunction with SC20. November 11, 2020

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Quantum characterization

Goal:

Quantify & understand the noise (errors) in a quantum processors.

Why?

- Quantum processors are small and too noisy to perform useful algorithms.
- We ultimately want to perform useful computation.

How?

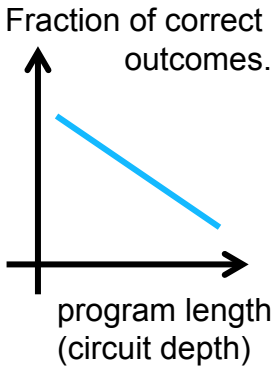
- **Benchmarking:** Assess the **overall performance** of a quantum processor.
- **Model-based characterization**: Create a **detailed error model** capable of predicting the observed processor output. Insights into specific error mechanisms can ideally be gained by looking at the model.

Quantum characterization (in more detail)

Benchmarking

Assess the **overall performance** of a quantum processor.

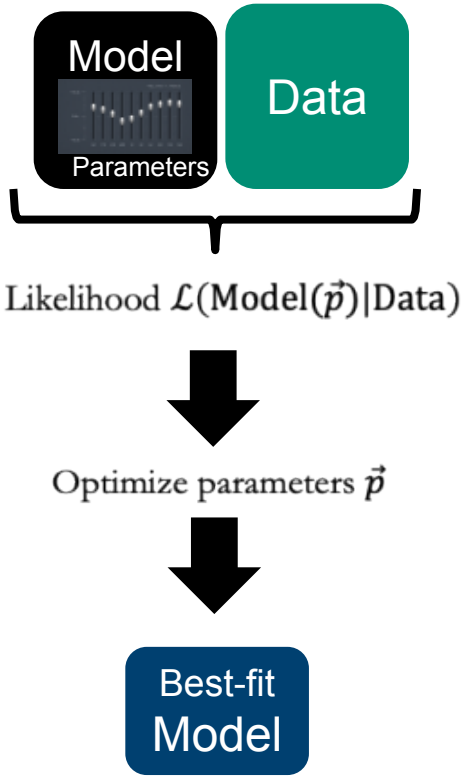
- Execute random circuits of different depths and track probability of getting the correct outcome.
- Example protocols:
  - Clifford randomized benchmarking (RB)
  - direct RB
  - mirror RB
  - volumetric benchmarking



Model-based characterization

Create a **detailed error model** capable of predicting the observed processor output.

- A parameterized noise model is fit to experimental data, often from executing *structured* circuits, resulting in a best-fit model.
- Example protocols:
  - gate set tomography (the “GST” in pyGSTi)
  - reduced-model tomography
  - idle tomography



Why does characterization warrant “Software”?

(software packages and products devoted solely to quantum characterization)

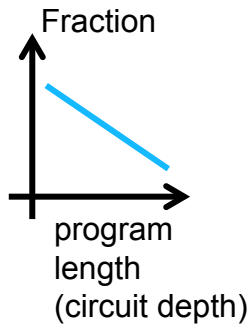
Generally applicable:

- Largely hardware agnostic
- Works within paradigm of executing programs (circuits) on a quantum computer

Characterization methods contain technical challenges:

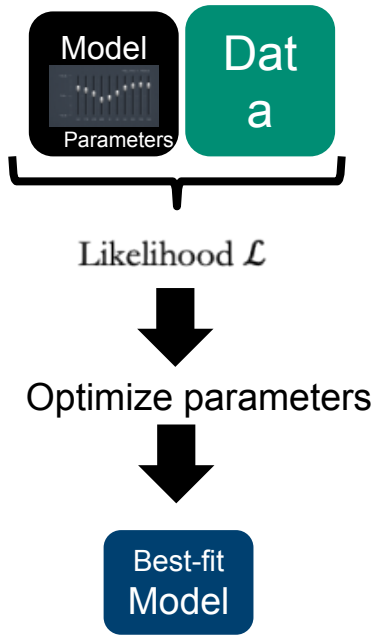
Benchmarking

- **Choosing a circuit family**
  - result should be meaningful
  - circuits must have known oucome
- **Sampling from the circuit distribution**
  - may be difficult to sample from
  - e.g. 2-qubit Cliffords.



Model-based characterization

- **Choosing meaningful and efficient models** (describe data but not too many parameters)
- **Constructing structured circuits** that are sensitive to model parameters.
- **Simulating circuits** (exponential in qubit #)
- **Optimization** over model space (a non-convex, many-parameter optimization)



Objections

Don’t the physicists in the lab know the best way to improve their hardware?

Yes and no. We’ve found that while experimentalists have great intuition and are often aware of some (if not all) of the major sources of noise in their system, characterization protocols offer more than just a “sanity check” and **complement experimentalists’ knowledge**. This is especially true regarding more complex many-qubit devices.

Can’t we just use techniques to randomize noise so that detailed noise models are irrelevant?

Techniques such as randomized compiling can be used to essentially turn many types of noise into depolarizing noise, which is likely to be more amenable to error correction. We view this approach as a **last resort**, however, as it can unnecessarily convert noise that could have been otherwise compensated for.

pygsti - a toolkit for quantum characterization

- ~ 150K lines of **Python** code.
- Implements a number of hardware-agnostic, general-use **characterization protocols**
- **Open source** (**Apache 2.0** license) on GitHub: www.pygsti.info

- **Well-documented**
  - Tutorials
  - Online documentation at pygsti.readthedocs.io
- **Contributions welcome**
  - contributors must sign a contributor license agreement
- **Research code**
  - contributed to by physicists (& software engineers)
  - contains cutting-edge techniques developed by Sandia's QPL.
  - opportunities for collaboration

Protocols & functionality in pygsti

Circuit (& data) simulation

Model testing

Metrics (process fidelity, diamond norm)

Gate Set Tomography (GST)
- 1- and 2-qubit
- Reduced model (allows more qubits)

Randomized Benchmarking (RB)
- Clifford (1- and 2-qubit)
- Direct (up to ~5 qubits)
- Mirror (5+ qubits)

Volumetric Benchmarking

Robust Phase Estimation (RPE) (1+ qubits)

Idle Tomography (crosstalk, up to 10 qubits)

Context-dependence detection (e.g. crosstalk & drift)

Time-dependent noise characterization

Reports (text, PDF, HTML)

Publications

Publications primarily about pyGSTi:

- Erik Nielsen *et al, "*Probing quantum processor performance with `pyGSTi`" 2020 *Quantum Sci. Technol.* **5** 044002

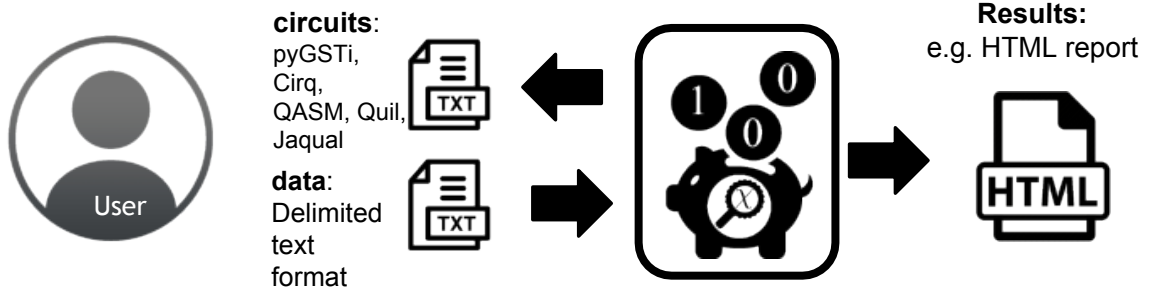
Publications using pyGSTi to perform experimental research:

- **R. Blume-Kohout**, J. K. Gamble, E. Nielsen, K. Rudinger, J. Mizrahi, K. Fortier, and P. Maunz, Nat. Commun. 8, 14485 (2017).
- J. P. Dehollain, J. T. Muhonen, R. Blume-Kohout, K. M. Rudinger, J. K. Gamble, E. Nielsen, A. Laucht, S. Sim- mons, R. Kalra, A. S. Dzurak, et al., New J. Phys. 18, 103018 (2016).
- **T. J. Proctor**, A. Carignan-Dugas, K. Rudinger, E. Nielsen, R. Blume-Kohout, and K. Young, Phys. Rev. Lett. 123, 030503 (2019).
- D. Kim, D. Ward, C. Simmons, J. K. Gamble, R. Blume- Kohout, E. Nielsen, D. Savage, M. Lagally, M. Friesen, S. Coppersmith, et al., Nat. Nanotechnol. 10, 243 (2015).
- K. Rudinger, S. Kimmel, D. Lobser, and P. Maunz, Phys. Rev. Lett. 118, 190502 (2017).
- M. Rol, C. Bultink, T. O'Brien, S. de Jong, L. Theis,
- X. Fu, F. Luthi, R. Vermeulen, J. de Sterke, A. Bruno, et al., Phys. Rev. Appl. 7, 041001 (2017)
- **K. Rudinger**, T. Proctor, D. Langharst, M. Sarovar, K. Young, and R. Blume-Kohout, Phys. Rev. X 9, 021045 (2019).
- S. Mavadia, C. L. Edmunds, C. Hempel, H. Ball, F. Roy, T. M. Stace, and M. J. Biercuk, npj Quantum Informa- tion 4, 7 (2018).
- M. Ware, G. Ribeill, D. Riste, C. A. Ryan, B. John- son, and M. P. da Silva, arXiv preprint arXiv:1803.01818 (2018).
- **T. Proctor**, M. Revelle, E. Nielsen, K. Rudinger, D. Lobser, P. Maunz, R. Blume-Kohout, and K. Young, arXiv preprint arXiv:1907.13608 (2019).
- G. A. L. White, C. D. Hill, and L. C. L. Hollenberg, "Performance optimisation for drift-robust fidelity improvement of two-qubit gates," (2019), arXiv:1911.12096 [quant-ph].
- Y. Chen, M. Farahzad, S. Yoo, and T.-C. Wei, Phys. Rev. A 100, 052315 (2019).
- **M. Sarovar**, T. Proctor, K. Rudinger, K. Young, E. Nielsen, and R. Blume-Kohout, "Detecting crosstalk errors in quantum information processors," (2019), arXiv:1908.09855 [quant-ph].
- **T. L. Scholten**, Y.-K. Liu, K. Young, and R. Blume- Kohout, "Classifying single-qubit noise using machine learning," (2019), arXiv:1908.11762 [quant-ph].
- L. C. G. Govia, G. J. Ribeill, D. Rist`e, M. Ware, and H. Krovi, "Bootstrapping quantum process tomography via a perturbative ansatz," (2019), arXiv:1902.10821 [quant-ph].
- S. S. Hong, A. T. Papageorge, P. Sivarajah, G. Crossman, N. Didier, A. M. Polloreno, E. A. Sete, S. W. Turkowski, M. P. da Silva, and B. R. Johnson, Phys. Rev. A 101, 012302 (2020).
- M. R. Geller, "Rigorous measurement error correction," (2020), arXiv:2002.01471 [quant-ph].
- M. K. Joshi, A. Elben, B. Vermersch, T. Brydges, C. Maier, P. Zoller, R. Blatt, and C. F. Roos, "Quantum information scrambling in a trapped-ion quantum simulator with tunable range interactions," (2020), arXiv:2001.02176 [quant-ph].

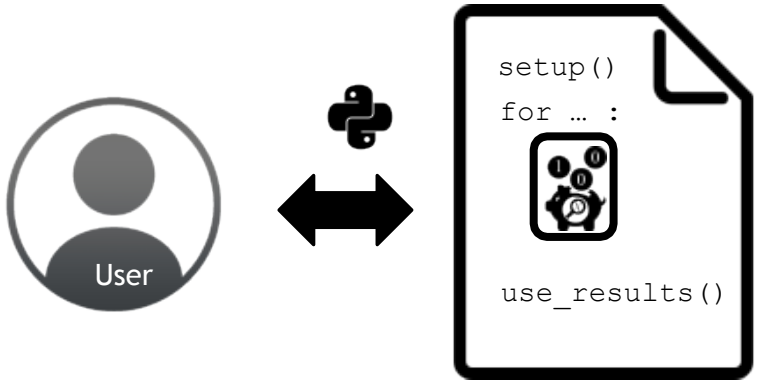
How is quantum characterization software be used in practice?

We've seen pyGSTi incorporated into user workflows in 3 primary ways :

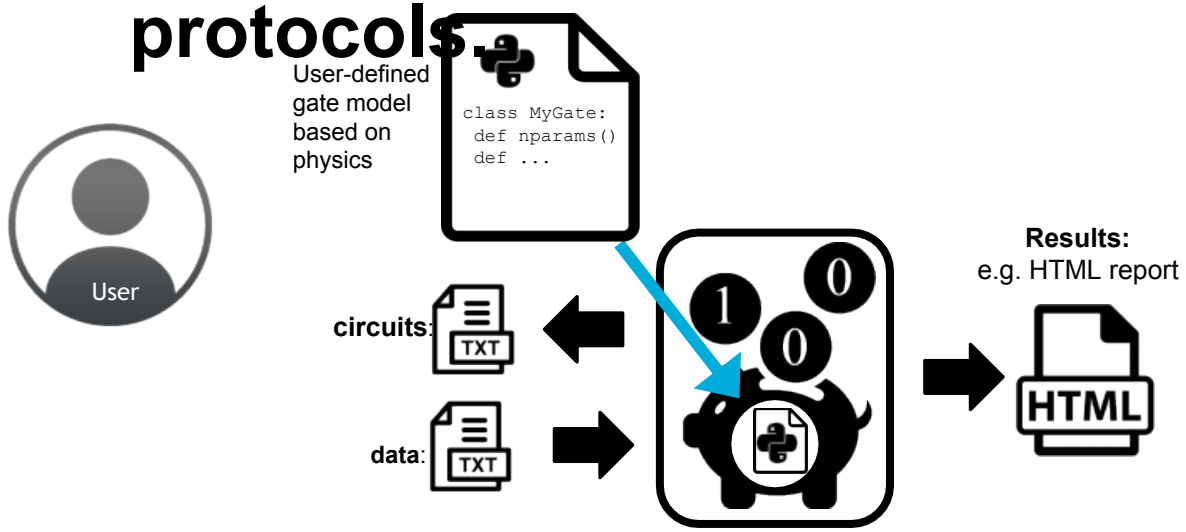
1. As a standalone QCVV tool



2. Within a Python analysis toolchain.



3. Customized QCVV protocols.



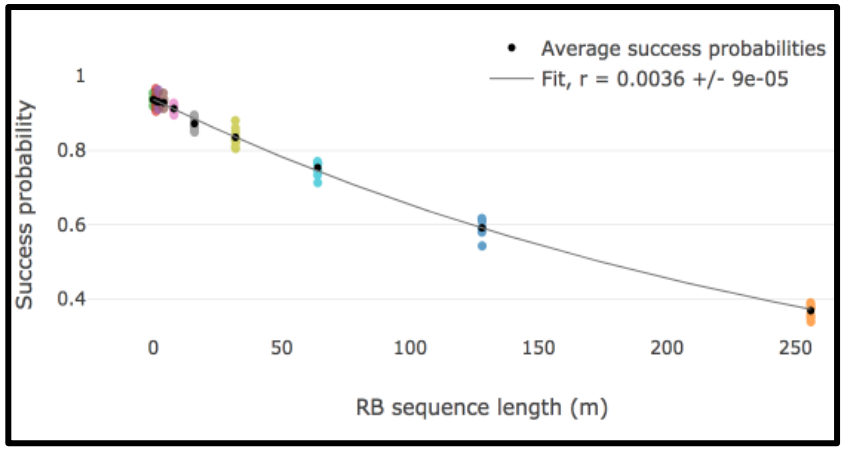
Two specific examples

Direct RB on 4 qubits:

```
protocol = pygsti.protocols.RB()
design = pygsti.protocols.DirectRBDesign(pspec,
    depths=[0, 1, 2, 4, 8, 16, 32, 64, 128, 256],
    circuits_per_depth=10,
qubit_labels=['Q0','Q1','Q2', 'Q3'])

pygsti.io.write_empty_protocol_data(design, 'example_rb')
# --- TAKE DATA HERE (fill in dataset.txt) ---
data = pygsti.io.load_data_from_dir('example_rb')
results = protocol.run(data)

ws = pygsti.report.Workspace()
ws.init_notebook_mode(autodisplay=True)
ws.RandomizedBenchmarkingPlot(results)
```



Gate set tomography on 1 qubit:

```
1  protocol = pygsti.protocols.StandardGST('TP,CPTP,Target')
2  design = pygsti.protocols.StandardGSTDesign(smq1Q_XYI.target_model(),
3      smq1Q_XYI.prep_fiducials(), smq1Q_XYI.meas_fiducials(),
4  smq1Q_XYI.germs(), max_lengths=[1,2,4,8,16,32])
5
6  pygsti.io.write_empty_protocol_data(design, 'example_gst')
7  # --- TAKE DATA HERE (fill in dataset.txt) ---
8  data = pygsti.io.load_data_from_dir('example_gst')
9  results = protocol.run(data)
10
11 report = pygsti.report.construct_standard_report(
12     results, title="GST Tutorial Example Report", verbosity=2)
13 report.write_html("GSTExampleReport.html", single_file=True, verbosity=2)
14
```

Estimate
TP

G-Opt Param	Value
0: Metric for gate-to-target	frobenius
0: Metric for SPAM-to-target	frobenius
0: Item weights	gates=1, spam=1

GST Tutorial Example Report

generated by pyGSTi on September 16, 2020

Summary

Welcome to a pyGSTi analysis report! This report is organized into "tabs", each of which is accessible from the sidebar on the left. This Summary tab summarizes the most popular analyses and figures of merit. Much more detailed analysis is available on other tabs. If this report encapsulates multiple datasets, estimates, or gauges, then you can switch between those using the dropdown menus on the sidebar. For more information about how to use this report, click on the Help tab link to the left.

Figure 1. Model violation summary.

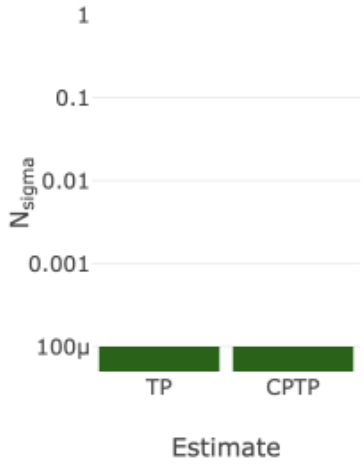


Figure 2. Model violation per iteration.

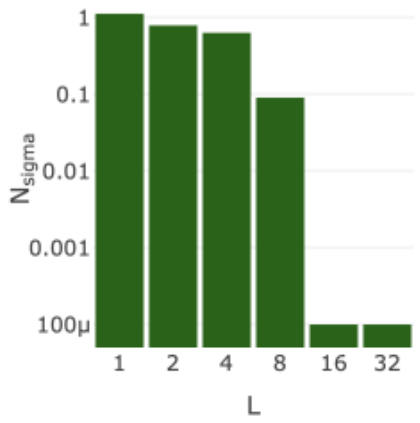


Figure 3. Histogram of per-circuit model violation.

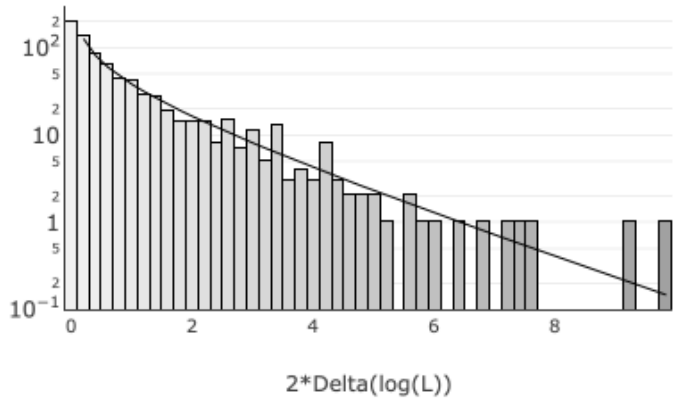


Table 1. Comparison of estimated gates to targets.

Gate	Entanglement Infidelity	1/2 Trace Distance	1/2 Diamond-Dist	Eigenvalue Ent. Infidelity	Eigenvalue 1/2 Diamond-Dist
[]	0.007544	0.007552	0.007553	0.007544	0.007696
Gxpi2:0	0.007459	0.007474	0.007479	0.007459	0.007522
Gypi2:0	0.007668	0.007687	0.007691	0.007668	0.007706

How has pyGSTi helped to debug and correct noise in QIPs?

See the narratives in published works, e.g.,

- R. Blume-Kohout, J. K. Gamble, E. Nielsen, K. Rudinger, J. Mizrahi, K. Fortier, and P. Maunz, Nat. Commun. 8, 14485 (2017).
- J. P. Dehollain, et al., New J. Phys. 18, 103018 (2016).
- D. Kim, et al., Nat. Nanotechnol. 10, 243 (2015).
- G. A. L. White, C. D. Hill, and L. C. L. Hollenberg, arXiv:1911.12096 [quant-ph] (2019).

Identification and fixing of “easy” errors along the way to a published result

- Errors in idle gate timing
- Errors in dynamical decoupling pulses
- Compiling errors
- Drift (e.g., loss of laser lock)

Current investigations:

- crosstalk in many-qubit (4-10 qubit) systems
- coherent gate errors in many-qubit systems

Summary and future of pyGSTi

Summary

- `pygsti` is an **open-source toolkit for characterizing quantum processors**.
- It implements a variety of characterization **protocols**.
- It contains nontrivial implementations of **fundamental routines** for data-fitting and statistical analysis.
- It is **customizable**, and offers a rich framework for creating noise models.

Outlook

- `pygsti` continues to be under active development.
- Development focuses on
  - implementing **new protocols and features**, specifically for characterization of larger systems
  - effort to **mature the codebase**. Goal of reaching version 1.0, marked by increased testing and a more stable API.
- There is a desire to integrate more with other existing tools (particularly in the area of circuit simulation);

Thanks for your attention!

- Relvant URLs:
  - pyGSTi on GitHub: www.pygsti.info
  - Documentation: pygsti.readthedocs.io
  - Quantum Performance Lab: qpl.sandia.gov
- pyGSTi Developers:
  - Erik Nielsen
  - Timothy Proctor
  - Kenneth Rudinger
  - Antonio Russo
  - Kevin Young
  - Robin Blume-Kohout
  - Robert Kelly
- Direct questions and comments to: pygsti@sandia.gov