JSON Input for Noise Modeling

# Summary

This document describes a general JSON schema that we'll be using to model quantum computing noise processes.

* We model gate noises as a set of channels, specifying by their Kraus operator matrices, that **follows** the logical operation of the specified gate.
* The list of Kraus matrices for each noise channel must satisfy the completely positive and trace-preserving (CPTP) condition.
* Each noise model JSON adheres to a configurable MSB/LSB bit-ordering scheme, which dictates the way to interpret multi-qubit noise channel Kraus operator matrices.
* Gates are identified by name and qubit operands.
* By default, noise Kraus operators will be applied to the same set of qubit operands as the quantum gate; hence their matrix dimensions must match the arity of the gate.
* Non-local noise Kraus operators, which act on qubits beyond gate operands, can be described by explicitly providing a list of qubits along with the matrix data in the Kraus operator JSON.
* Readout errors (per-qubit) are captured by two error probabilities: prob\_meas0\_prep1 (P(0|1)) and prob\_meas1\_prep0 (P(1|0)).

# Schema

Top-level **NoiseModel** is a JSON object which contains a list of **GateNoise** and **ReadoutError** objects and information about the MSB/LSB bit-order convention.

Table 1 - NoiseModel JSON field descriptions

|  |  |  |
| --- | --- | --- |
| Field Name | Descriptions | Type |
| bit\_order | MSB or LSB convention in which Kraus operator matrices are specified. | String |
| gate\_noise | List of noise specifications of quantum gates. | Array[GateNoise] |
| readout\_errors | List of readout error specifications for qubits. | Array[ReadoutError] |

Each **GateNoise** object contains information about the gate name, qubit operands, and the list of Kraus operators.

Table 2 - GateNoise JSON field descriptions

|  |  |  |
| --- | --- | --- |
| Field Name | Descriptions | Type |
| gate\_name | Name of the gate (e.g. X, Y, Z, CNOT, etc.) | String |
| register\_location | Qubit operand labels | Array[String] |
| noise\_channels | Noise channels that will be applied *after* the logical operation of the specified gate | Array[ChannelKrausOp] |

Each **ChannelKrausOp** object contains a list of matrices representing channel Kraus operators.

These matrices must satisfy the CPTP condition, i.e.

where is the list of Kraus matrices provided in each **ChannelKrausOp**.

For multi-qubit operators, the matrix shall be described in the bit-order (MSB/LSB) convention of this **NoiseModel** JSON.

The optional *noise\_qubits* field can be used to explicitly specify the qubits to which the Kraus term will be applied. If none provided, the Kraus term will be applied to the gate qubit operands.

Table 3 - KrausOp JSON field descriptions

|  |  |  |
| --- | --- | --- |
| Field Name | Descriptions | Type |
| matrix | List of Kraus matrices. | Array[Matrix[Complex]] |
| noise\_qubits | (Optional) Qubits to which this Kraus operator is applied.  If not provided, this Kraus operator will be applied to the gate qubit operands (register\_location field). | Array[String] |

Notes:

* Matrix type is serialized row-by-row, i.e., as a list of row arrays.
* Complex type is serialized as a pair of real and imaginary values.

Each **ReadoutError** object contains information about the qubit location and probabilities of incorrect measurements.

Table - ReadoutError field descriptions

|  |  |  |
| --- | --- | --- |
| Field Name | Descriptions | Type |
| register\_location | Qubit label | String |
| prob\_meas0\_prep1 | Probability to read out 0 while the actual result (e.g., unbiased sampling of the state-vector) should be 1. | Double |
| prob\_meas1\_prep0 | Probability to read out 1 while the actual result (e.g., unbiased sampling of the state-vector) should be 0. | Double |

The above descriptions are summarized in the below JSON Schema:

{

"type": "object",

"title": "NoiseModel",

"description": "Noise Model input for XACC",

"properties": {

"bit\_order": {

"type": "string",

"description": "MSB or LSB bit-order"

},

"readout\_errors": {

"type": "array",

"description": "List of readout error specifications",

"items": {

"type": "object",

"title": "ReadoutError",

"properties": {

"register\_location": {

"type": "string",

"description": "Readout qubit"

},

"prob\_meas0\_prep1": {

"type": "double",

"description": "P(0|1)"

},

"prob\_meas1\_prep0": {

"type": "double",

"description": "P(1|0)"

}

}

}

},

"gate\_noise": {

"type": "array",

"description": "List of gate noise specifications",

"items": {

"type": "object",

"title": "GateNoise",

"properties": {

"gate\_name": {

"type": "string",

"description": "Gate name"

},

"register\_location": {

"type": "array[string]",

"description": "Gate qubit operands"

},

"noise\_channels": {

"type": "array",

"items": {

"type": "object",

"title": "ChannelKrausOp",

"properties": {

"matrix": {

"type": "ndarray",

"description": "List of channel Kraus matrices"

},

"noise\_qubits": {

"type": "array[string]",

"description": "Kraus op qubits",

"optional": True

}

}

}

}

}

}

}

}

}

Examples of noise model generation (in Python)

See IPython notebook (WIP)

<https://code.ornl.gov/7tn/noise-model/-/blob/master/Noise_Json.ipynb>

# Sample JSON

\_\_TBD\_\_

# Python Class Structure

\_\_TBD\_\_

# Examples

\_\_TBD\_\_

Providing example of generating Kraus matrices (both LSB and MSB cases) and JSON for amplitude damping, dephasing, and depolarization noise channels.