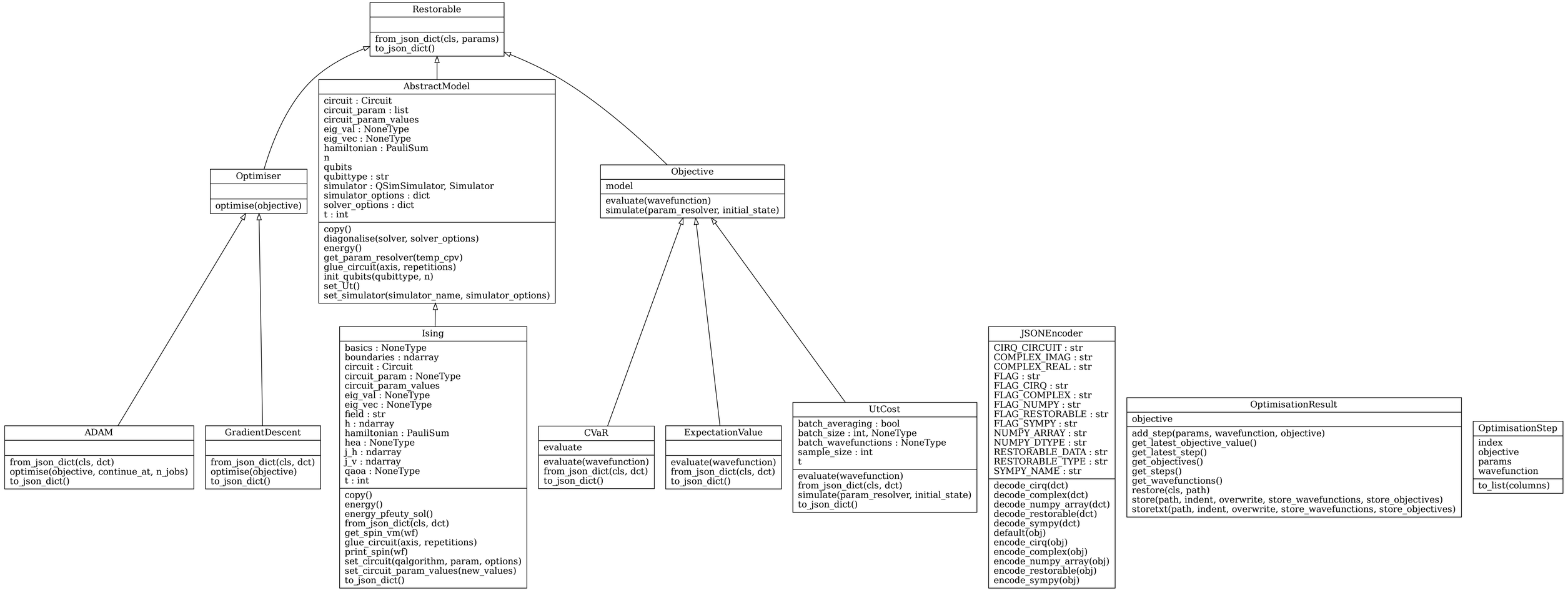
Restorable
from_json_dict(cls, params)
to_json_dict()
AbstractModel
circuit : Circuit
circuit_param : list
circuit_param_values
eig_val : NoneType
eig_vec : NoneType
hamiltonian : PauliSum
n
qubits
qubittype : str
simulator : QSimSimulator, Simulator
simulator_options : dict
solver_options : dict
t : int
copy()
diagonalise(solver, solver_options)
energy()
get_param_resolver(temp_cpv)
glue_circuit(axis, repetitions)
init_qubits(qubittype, n)
set_Ut()
set_simulator(simulator_name, simulator_options)
Optimiser
optimise(objective)
Objective
model
evaluate(wavefunction)
simulate(param_resolver, initial_state)
Ising
basics : NoneType
boundaries : ndarray
circuit : Circuit
circuit_param : NoneType
circuit_param_values
eig_val : NoneType
eig_vec : NoneType
field : str
h : ndarray
hamiltonian : PauliSum
hea : NoneType
j_h : ndarray
j_v : ndarray
qaoa : NoneType
t : int
copy()
energy()
energy_pfeuty_sol()
from_json_dict(cls, dct)
get_spin_vm(wf)
glue_circuit(axis, repetitions)
print_spin(wf)
set_circuit(qalgorithm, param, options)
set_circuit_param_values(new_values)
to_json_dict()
ADAM
from_json_dict(cls, dct)
optimise(objective, continue_at, n_jobs)
to_json_dict()
GradientDescent
from_json_dict(cls, dct)
optimise(objective)
to_json_dict()
CVaR
evaluate
evaluate(wavefunction)
from_json_dict(cls, dct)
to_json_dict()
ExpectationValue
evaluate(wavefunction)
from_json_dict(cls, dct)
to_json_dict()
UtCost
batch_averaging : bool
batch_size : int, NoneType
batch_wavefunctions : NoneType
sample_size : int
t
evaluate(wavefunction)
from_json_dict(cls, dct)
simulate(param_resolver, initial_state)
to_json_dict()
JSONEncoder
CIRQ_CIRCUIT : str
COMPLEX_IMAG : str
COMPLEX_REAL : str
FLAG : str
FLAG_CIRQ : str
FLAG_COMPLEX : str
FLAG_NUMPY : str
FLAG_RESTORABLE : str
FLAG_SYMPY : str
NUMPY_ARRAY : str
NUMPY_DTYPE : str
RESTORABLE_DATA : str
RESTORABLE_TYPE : str
SYMPY_NAME : str
decode_cirq(dct)
decode_complex(dct)
decode_numpy_array(dct)
decode_restorable(dct)
decode_sympy(dct)
default(obj)
encode_cirq(obj)
encode_complex(obj)
encode_numpy_array(obj)
encode_restorable(obj)
encode_sympy(obj)
OptimisationResult
objective
add_step(params, wavefunction, objective)
get_latest_objective_value()
get_latest_step()
get_objectives()
get_steps()
get_wavefunctions()
restore(cls, path)
store(path, indent, overwrite, store_wavefunctions, store_objectives)
storetxt(path, indent, overwrite, store_wavefunctions, store_objectives)
OptimisationStep
index
objective
params
wavefunction
to_list(columns)