UNIVERSITY OF TORONTO

Designing Quantum Algorithms with Tequila

Jakob Kottmann (jakob.kottmann@utoronto.ca)

Sumner Alperin-Lea, Teresa Tamayo-Mendoza, Cyrille Lavigne, Abhinav Anand, Alba Cervera-Lierta

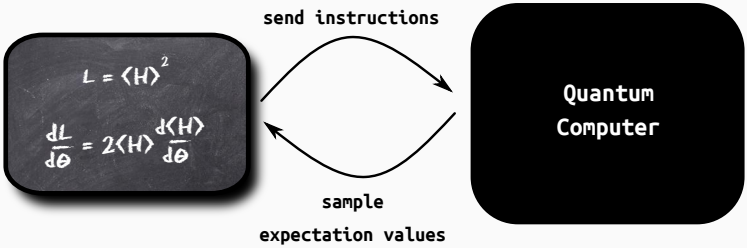
Alán Aspuru-Guzik

Matter Group Toronto

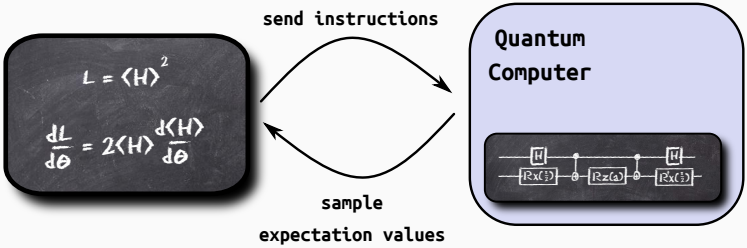
May 27, 2020

https://github.com/aspuru-guzik-group/tequila

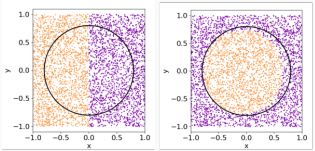
Introduction



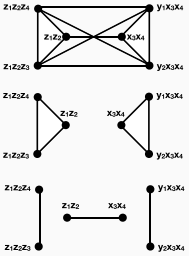
Introduction



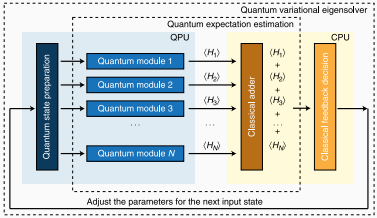
Introduction



A. Pérez-Salinas *et. al*, Quantum 4, 226 (2020)



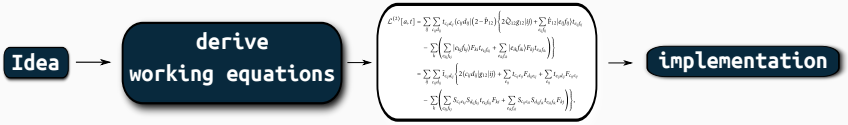
T.C. Yen *et. al*, J. Chem. Theory Comput. 2020, 16, 4, 2400–2409



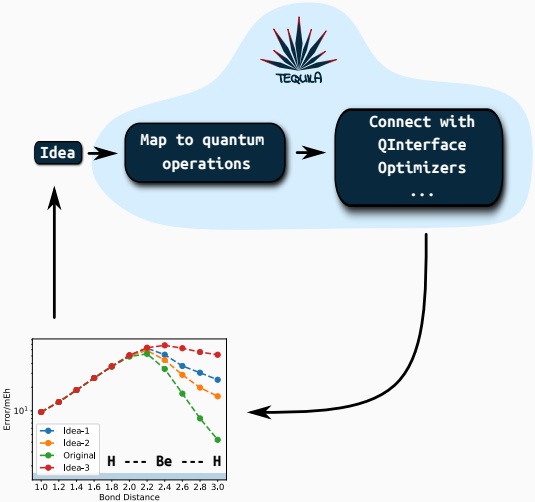
A. Peruzzo, J. McClean *et. al.* Nature Comm. 5, 4213 (2014)

Introduction

Typical Quantum Chemistry Development Workflow



Introduction



Tequila: General Idea

Tequila: General Idea

$$G = |\Psi\rangle\langle 001| \qquad |\Psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$

$$U = e^{-i\frac{\theta}{2}(G + G^\dagger)} \qquad H = |\Psi\rangle\langle\Psi| - Z(0)Z(1)$$

$$L = \min \langle H \rangle_U^2$$

Inspired by:

SIAM J. Sci. Comp, 2016; https://github.com/m-a-d-n-e-s-s

Tequila: General Idea

$$G = |\Psi\rangle\langle 00| \qquad |\Psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$

$$U = e^{-i\frac{\theta}{2}(G + G^\dagger)} \qquad H = |\Psi\rangle\langle\Psi| - Z(0)Z(1)$$

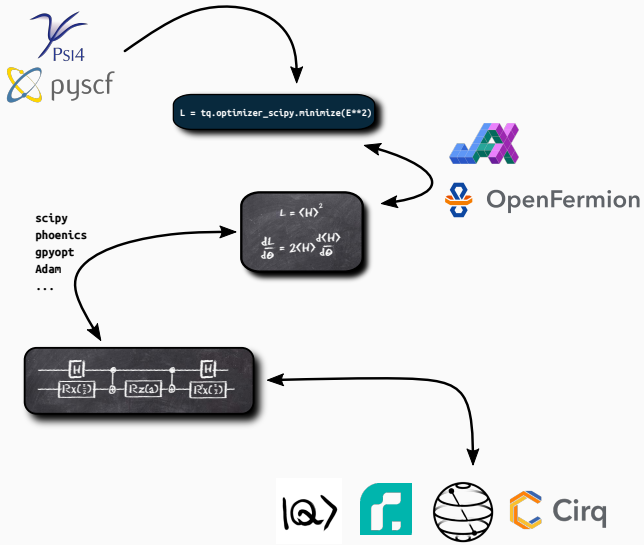
$$L = \min \langle H \rangle_U^2$$

```
wfn = tq.QubitWaveFunction.from_string("1.0|01> + 1.0|10>")
wfn = wfn.normalize()
G = tq.paulis.KetBra(ket=wfn, bra="1.0*|00>")
U = tq.gates.Trotterized(G + G.dagger(), angle="a", steps=1)
H = tq.paulis.Projector(wfn)
expval = tq.ExpectationValue(H=H, U=U)
result = tq.minimize("bfgs", expval**2)
```

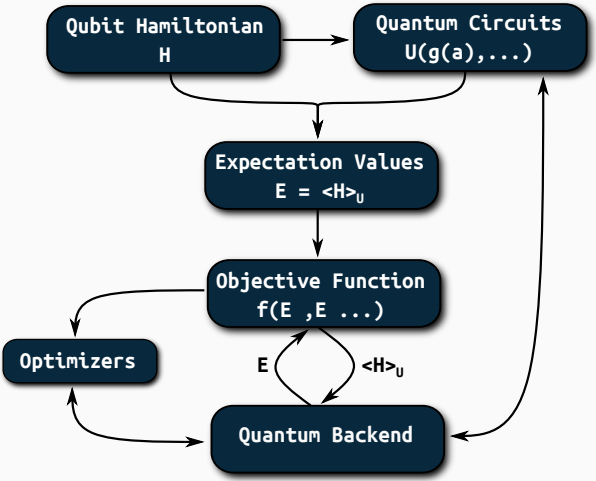
Inspired by:

SIAM J. Sci. Comp, 2016; https://github.com/m-a-d-n-e-s-s

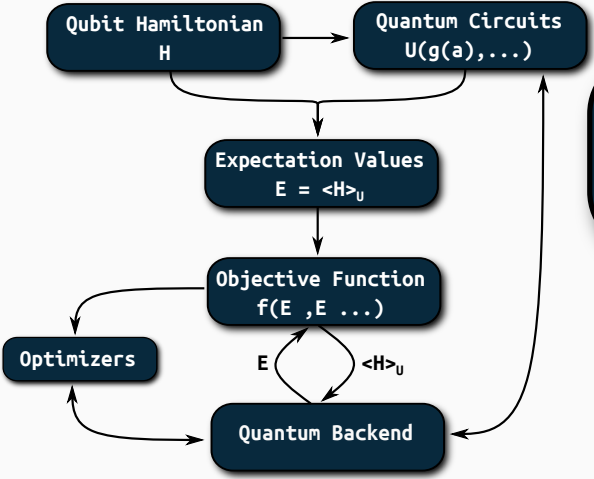
Tequila: General Idea



Tequila: API

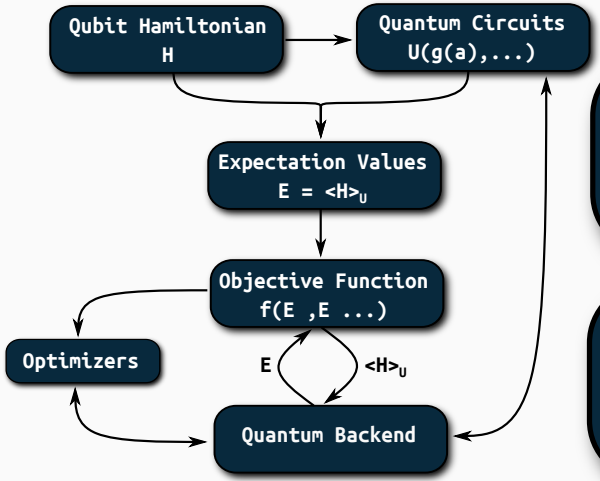

Qubit Hamiltonian
H
Quantum Circuits
U(g(a),...)
Expectation Values
E = <H>U
Objective Function
f(E ,E ...)
Optimizers
E
<H>U
Quantum Backend

Tequila: API




```
H = tq.paulis.X(0)
a = tq.Variable("a")
U = tq.gates.Ry(angle=a*tq.numpy.pi, target=0)
E = tq.ExpectationValue(H=H,U=U)
result = tq.optimizer_scipy.minimize(E**2)
```

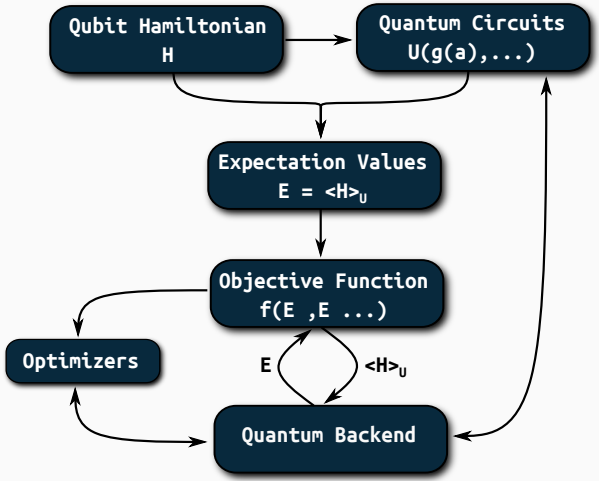
Tequila: API




```
H = tq.paulis.X(0)
a = tq.Variable("a")
U = tq.gates.Ry(angle=a*tq.numpy.pi, target=0)
E = tq.ExpectationValue(H=H,U=U)
result = tq.optimizer_scipy.minimize(E**2)
```

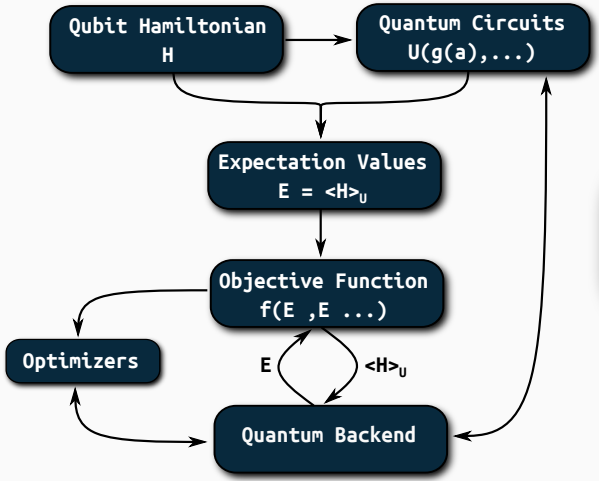
```
wfn = tq.simulate(U, variables={"a":1.0}
energy = tq.simulate(E, variables={"a":1.0})
dEda = tq.grad(E, "a")
egrad = tq.simulate(dEda, variables={"a":1.0})
d2Ed2a = tq.grad(dEda, "a")
```

Tequila: API




```
egrad = tq.simulate(dEda,
                    variables={"a":1.0},
                    backend="qiskit",
                    device="ibmq_rome")
```

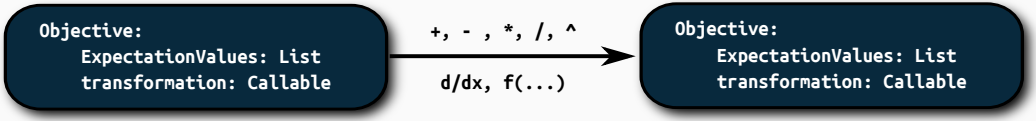
Tequila: API




```
mol = tq.Molecule("h2.xyz", "sto-3g", "JW")
H = mol.make_hamiltonian()
U = mol.prepare_reference()
G = mol.make_excitation_generator([(0,2),(1,3)])
U += tq.gates.Trotterized(generators=[G], angles=["a"])
```

A bit more Details

Operations


Objective:
ExpectationValues: List
transformation: Callable
+, - , *, /, ^
d/dx, f(...)
Objective:
ExpectationValues: List
transformation: Callable

Operations

```
O1 = E0 + E1
```

Operations

```
O1 = E0 + E1
```

```
Objective:
    ExpectationValues = [E0, E1]
    transformation = x+y
```

Operations

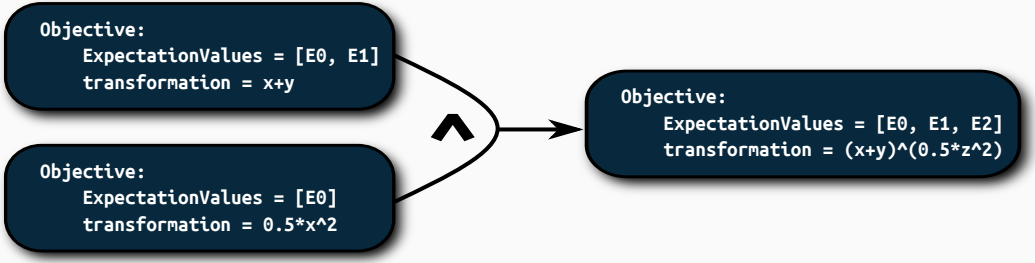
```
O1 = E0 + E1
O2 = 0.5*E0**2
```

```
Objective:
    ExpectationValues = [E0, E1]
    transformation = x+y
```

```
Objective:
    ExpectationValues = [E0]
    transformation = 0.5*x^2
```

Operations

```
O1 = E0 + E1
O2 = 0.5*E0**2
O3 = O1**O2
```



Derivatives

Rotations: Shift Rule + JAX

$$\frac{d}{da}\langle H\rangle_{U(a)} = f(a)\left(\langle H\rangle_{U(a+x)} - \langle H\rangle_{U(a-x)}\right)$$

Shift rule: M. Schuld *et. al.* Phys. Rev. A *2019*

github.com/XanaduAI/pennylane/

github.com/google/jax

Derivatives

Rotations: Shift Rule + JAX

$$\frac{d}{da}\langle H\rangle_{U(a)} = f(a)\left(\langle H\rangle_{U(a+x)} - \langle H\rangle_{U(a-x)}\right)$$

Others: Compiler + Shift Rule + Chain Rule + JAX

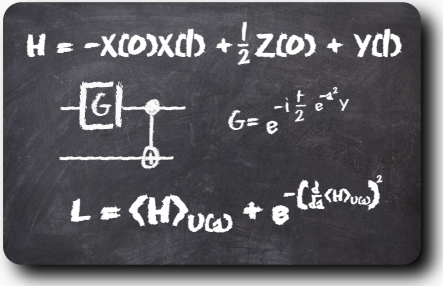
$$\frac{d}{da}\langle H\rangle_{U(a)} = \frac{d}{da}\langle H\rangle_{V(a)W(a)\ldots}$$

Shift rule: M. Schuld *et. al.* Phys. Rev. A *2019*

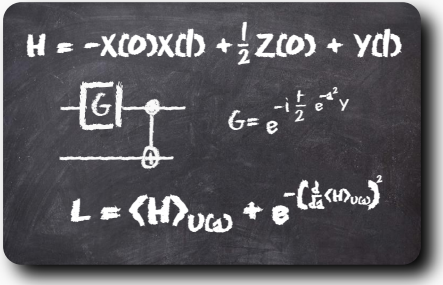
github.com/XanaduAI/pennylane/

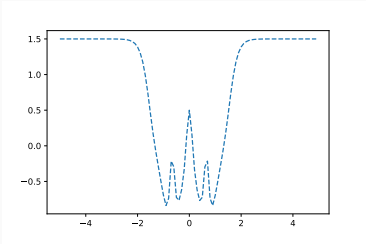
github.com/google/jax

Toy Example

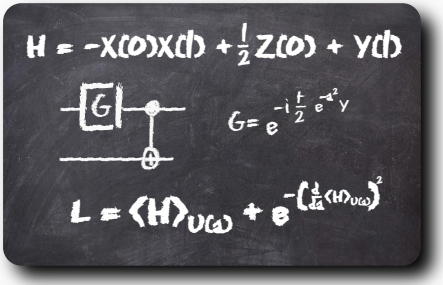


Toy Example

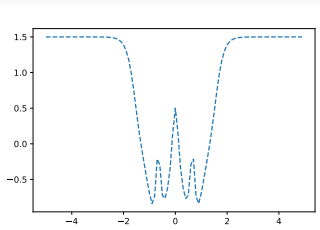




Toy Example



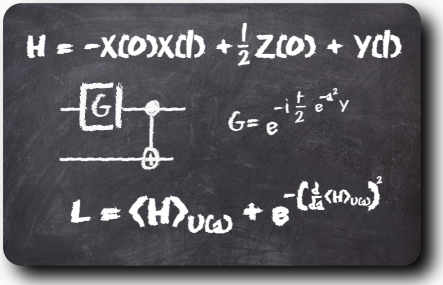

```
a = tq.Variable("a")
U = tq.gates.Ry(angle=(-a**2).apply(tq.numpy.exp)*pi, target=0)
U += tq.gates.X(target=1, control=0)
H = tq.QubitHamiltonian.from_string("-1.0*X(0)X(1)+0.5Z(0)+Y(1)")
E = tq.ExpectationValue(H=H, U=U)
dE = tq.grad(E, "a")
objective = E + (-dE**2).apply(tq.numpy.exp)
result = tq.minimize(method="phoenics", objective=objective)
```



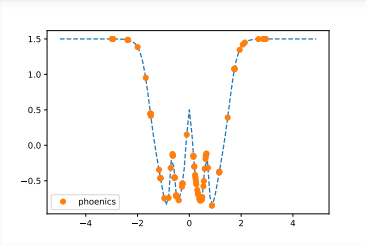
F. Häse *et. al*, ACS Cent. Sci. **2018**

https://github.com/aspuru-guzik-group/phoenics

Toy Example



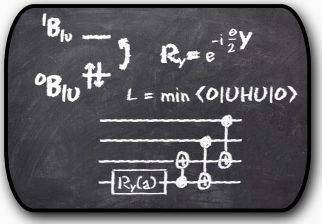

```
a = tq.Variable("a")
U = tq.gates.Ry(angle=(-a**2).apply(tq.numpy.exp)*pi, target=0)
U += tq.gates.X(target=1, control=0)
H = tq.QubitHamiltonian.from_string("-1.0*X(0)X(1)+0.5Z(0)+Y(1)")
E = tq.ExpectationValue(H=H, U=U)
dE = tq.grad(E, "a")
objective = E + (-dE**2).apply(tq.numpy.exp)
result = tq.minimize(method="phoenics", objective=objective)
```

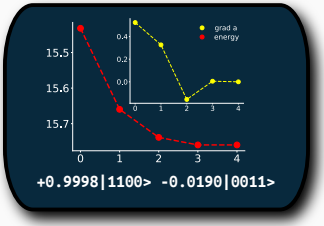


F. Häse *et. al*, ACS Cent. Sci. **2018**

https://github.com/aspuru-guzik-group/phoenics

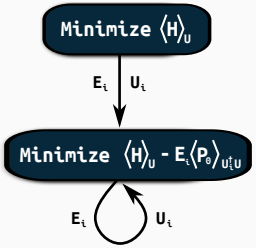
Some Examples for Chemistry

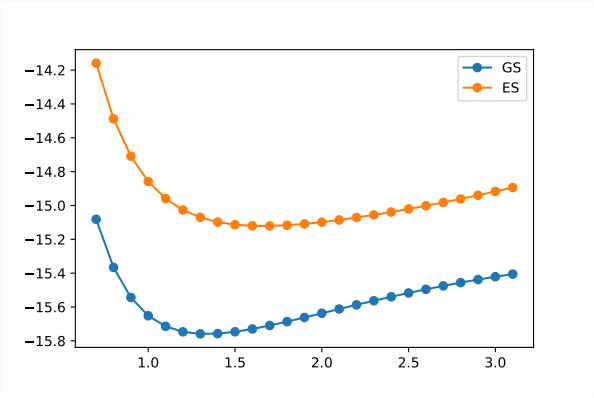





```
active = {"b1u": [0, 1]}
mol = tq.chemistry.Molecule("beh2.xyz", "6-31g", active)
H = mol.make_hamiltonian()
U = tq.gates.Ry("a", 0)
U += tq.gates.CNOT(0, 1) + tq.gates.CNOT(0, 2)
U += tq.gates.CNOT(1, 3) + tq.gates.X([2, 3])
expv = tq.ExpectationValue(U, H)
result = tq.optimizer_scipy.minimize(expv, "bfgs")
wfn = tq.simulate(U, variables=result.angles)
```

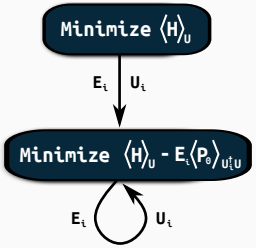
Some Examples for Chemistry





as used in: Lee, Huggings *et.al.* JCTC **2019**

Some Examples for Chemistry

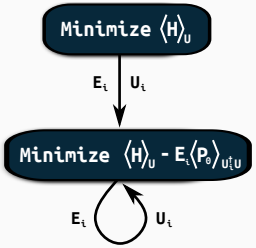


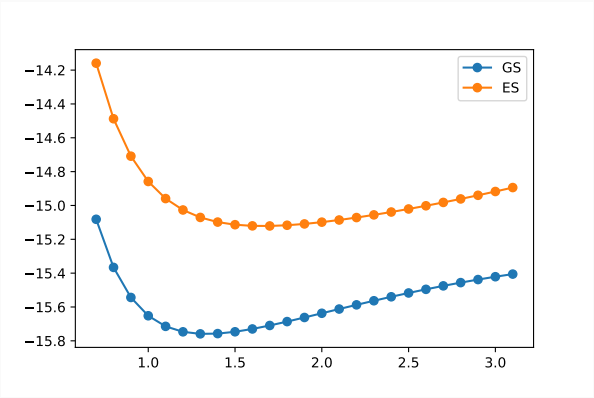

```
active = {"b1u": [0], "b2u": [0]}
mol = tq.chemistry.Molecule(geometry="data/beh2.xyz", basis_set="6-31g", active_orbitals=active)
H = mol.make_hamiltonian()
P0 = tq.paulis.Projector(0, n_qubits=4)
results = []
for i in range(2):
    U = tq.gates.Ry((i, "a"), 0)
    U += tq.gates.CNOT(0, 1) + tq.gates.CNOT(0, 2)
    U += tq.gates.CNOT(1, 3) + tq.gates.X([2, 3])
    E = tq.ExpectationValue(U, H)
    active_vars = E.extract_variables()
    angles = {angle: 0.0 for angle in active_vars}
    for data, U2 in results:
        S2 = tq.ExpectationValue(H=P0, U=U2.dagger() + U)
        E -= data.energy * S2
        angles = {**angles, **data.angles}

    result = tq.optimizer_scipy.minimize(E, method="bfgs", variables=active_vars, initial_values=angles)
    results.append((result, U))
```

as used in: Lee, Huggings *et.al.* JCTC **2019**

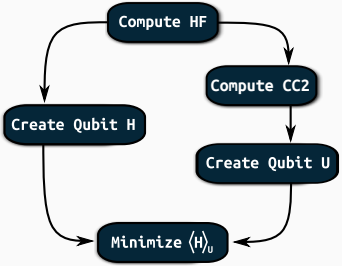
Some Examples for Chemistry



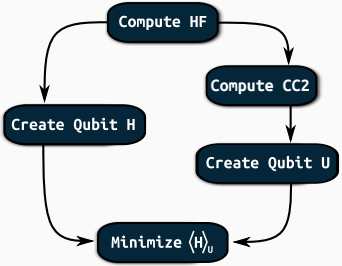


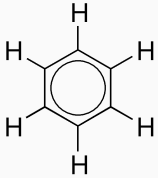
as used in: Lee, Huggings *et.al.* JCTC **2019**

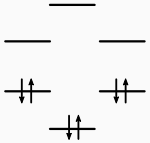
Some Examples for Chemistry: Ground State Optimization



Some Examples for Chemistry: Ground State Optimization

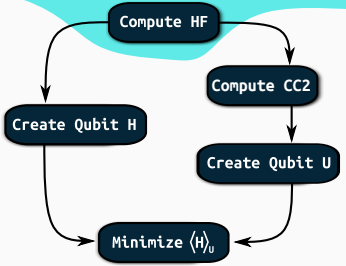


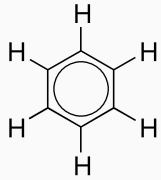


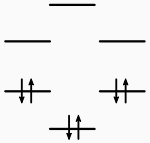


Some Examples for Chemistry: Ground State Optimization

```
active = {"B1u": [0], "B3g": [0, 1], "B2g": [0], "Au": [0], "b1u": [1]}
molecule = tq.quantumchemistry.Molecule(geometry="benzene.xyz", basis_set='sto-3g', active_orbitals=active)
```

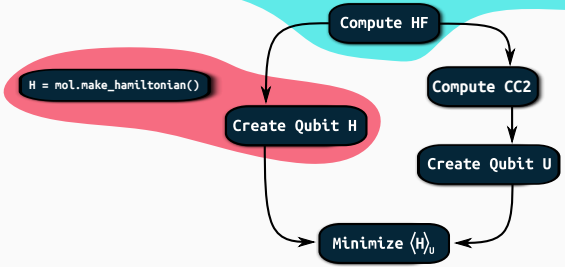


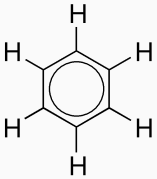


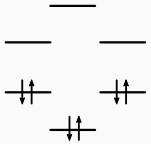


Some Examples for Chemistry: Ground State Optimization

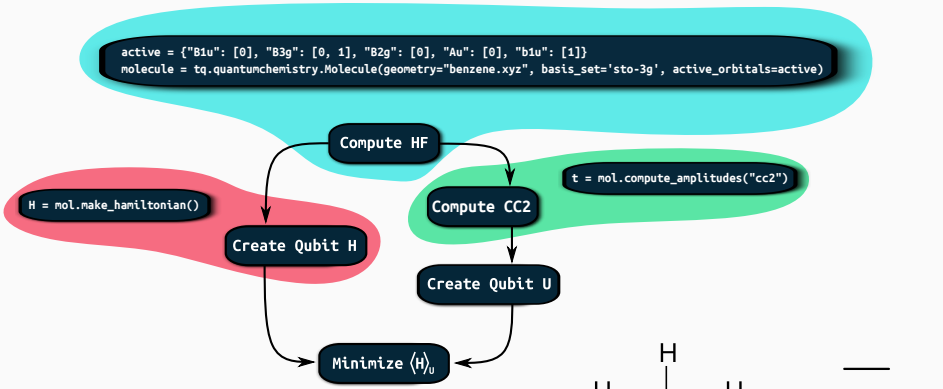
```
active = {"B1u": [0], "B3g": [0, 1], "B2g": [0], "Au": [0], "b1u": [1]}
molecule = tq.quantumchemistry.Molecule(geometry="benzene.xyz", basis_set='sto-3g', active_orbitals=active)
```



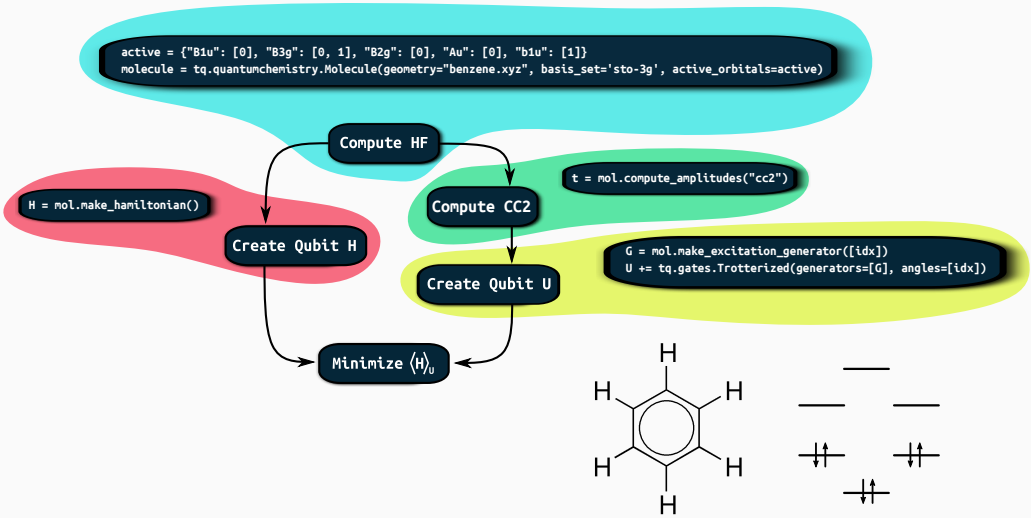




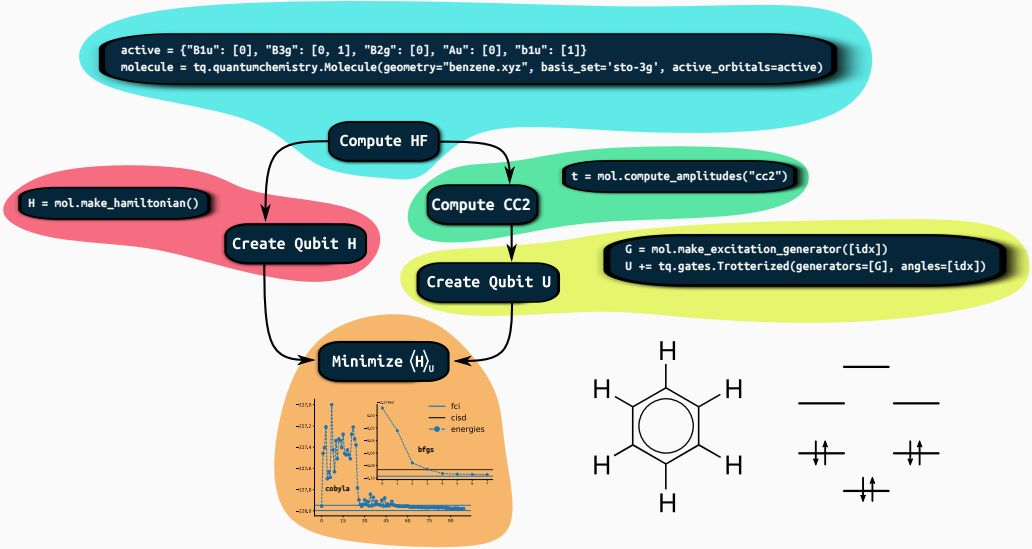
Some Examples for Chemistry: Ground State Optimization



Some Examples for Chemistry: Ground State Optimization



Some Examples for Chemistry: Ground State Optimization



Acknowledgement

Alán
Aspuru-Guzik

Sumner
Alperin-Lea

Teresa
Tamayo-Mendoza

Cyrille
Lavigne

Abhinav
Anand

Alba
Cervera-Lierta

Tequila Distillery:
- Skylar Chaney (Williamstown)
- Philipp Schleich (RHTW Aachen)
- Phillip Jensen
- Maha Kesebi
- Matthias Degroote
Izmaylov Group (UofT):
-Tzu Ching(Thomson) Yen
-Vladyslav Verteletskyi
- Robert Lang

Get the code from: https://github.com/aspuru-guzik-group/tequila