Continuous Variable Quantum Computing: CV-QC

Soronzonbold Otgonbaatar

DLR Oberpfaffenhofen
Star Wars: CV-QC
The field $E$ is the emergent physical property of many photons and creations.
CV-QC: Star Wars
CV-QC

\[ E_x(t, z) \approx A(t)B(z) \]
$|n\rangle$

$\hat{E}_x(z, t) \approx \hat{A}(t)B(z)$
CV-QC

\[ \langle n | \hat{E} | n \rangle \neq E_x(t, z) \]

What is \( |? \rangle \)?
CV-QC

\[ \langle \alpha | \hat{E} | \alpha \rangle = E_X(t, z) \]

Coherent state: \[ |\alpha\rangle \approx \sum_i f(n_i) \]
Photonic quantum computing

Classic photonic computing was here (for decades) even before photonic quantum computing. Photons are charge-less and spin 0. It does not interact with environment. Photons even donot interact with each other, and only via nonlinear optical medium.

Recently (around 2001), linear optical medium is discovered. Hence, they become promising candidates for building room temperature CV-QCs. No-self interaction, and so you do not see many papers for computing optimization problems.
Photonic quantum computing

~1980

Today-Future
Shut up and Calculate - „David Mermin“

Strawberry and Pennylane
Strawberry and Pennylane

As PQC and Operations

<table>
<thead>
<tr>
<th>Gates</th>
<th>As PQC and Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation(\phi, wires)</td>
<td>Phase space rotation</td>
</tr>
<tr>
<td>Squeezing(\gamma, wires)</td>
<td>Phase space squeezing</td>
</tr>
<tr>
<td>Displacement(\alpha, \phi, wires)</td>
<td>Phase space displacement</td>
</tr>
<tr>
<td>Beamsplitter(\theta_a, \phi, wires)</td>
<td>Beamsplitter interaction</td>
</tr>
<tr>
<td>TwoModeSqueezing(\phi, wires)</td>
<td>Phase space two-mode squeezing</td>
</tr>
<tr>
<td>QuadraticPhase(\alpha, wires)</td>
<td>Quadratic phase shift</td>
</tr>
<tr>
<td>ControlledAddition(\theta, wires)</td>
<td>Controlled addition operation</td>
</tr>
<tr>
<td>ControlledPhase(\phi, wires)</td>
<td>Controlled phase operation</td>
</tr>
<tr>
<td>Kerr(\kappa, wires)</td>
<td>Kerr interaction</td>
</tr>
<tr>
<td>CrossKerr(\kappa_a, \kappa_p, wires)</td>
<td>Cross-Kerr interaction</td>
</tr>
<tr>
<td>CubicPhase(\gamma, \kappa, wires)</td>
<td>Cubic phase shift</td>
</tr>
<tr>
<td>Interferometer(\Gamma, wires)</td>
<td>A linear interferometer transforming the bosonic operators according to the unitary matrix ( U )</td>
</tr>
</tbody>
</table>

State preparation  As encoding classical data

<table>
<thead>
<tr>
<th>State preparation</th>
<th>As encoding classical data</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoherentState(\alpha, \phi, wires)</td>
<td>Prepares a coherent state.</td>
</tr>
<tr>
<td>SqueezedState(\gamma, \phi, wires)</td>
<td>Prepares a squeezed vacuum state.</td>
</tr>
<tr>
<td>DisplacedSqueezedState(\alpha, \phi_a, \gamma, \phi_r, wires)</td>
<td>Prepares a displaced squeezed vacuum state.</td>
</tr>
<tr>
<td>ThermalState(\bar{n}, wires)</td>
<td>Prepares a thermal state.</td>
</tr>
<tr>
<td>GaussianState(\mu, \sigma, \gamma, \bar{n}, wires)</td>
<td>Prepares subsystems in a given Gaussian state.</td>
</tr>
<tr>
<td>FockState(\alpha, \phi, wires)</td>
<td>Prepares a single Fock state.</td>
</tr>
<tr>
<td>FockStateVector(\alpha, \phi, wires)</td>
<td>Prepares subsystems using the given ket vector in the Fock basis.</td>
</tr>
<tr>
<td>FockDensityMatrix(\rho, \gamma, \sigma, \bar{n}, wires)</td>
<td>Prepares subsystems using the given density matrix in the Fock basis.</td>
</tr>
<tr>
<td>CatState(\alpha, \phi, \gamma, \sigma, \bar{n}, wires)</td>
<td>Prepares a cat state.</td>
</tr>
</tbody>
</table>

Observables  As outputs of class label or functions

<table>
<thead>
<tr>
<th>Observables</th>
<th>As outputs of class label or functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumberOperator(wires)</td>
<td>The photon number observable ( \hat{n} ).</td>
</tr>
<tr>
<td>X(wires)</td>
<td>The position quadrature observable ( \hat{x} ).</td>
</tr>
<tr>
<td>P(wires)</td>
<td>The momentum quadrature observable ( \hat{p} ).</td>
</tr>
<tr>
<td>QuadOperator(\phi, wires)</td>
<td>The generalized quadrature observable ( \hat{z}_\phi = \hat{x}\cos\phi + \hat{p}\sin\phi ).</td>
</tr>
<tr>
<td>PolyXP(\gamma, \kappa, \phi, wires)</td>
<td>An arbitrary second-order polynomial observable.</td>
</tr>
<tr>
<td>FockStateProjector(\alpha, \phi, wires)</td>
<td>The number state observable (</td>
</tr>
</tbody>
</table>
Regression on CNN: $f(x) = \sin(x)$
Regression on CNN: sin(x) with sigmoid

\[
L = 1, \ z=10
\]

\[
L = \frac{1}{N} \sum_i (y_i - \tilde{y}_i)^2
\]

\[
y_i = \sin(x_i)
\]
Regression on CNN: Nonlinear functions at the output neuron

$\text{sigmoid}(y_i)$
Regression on CNN: \( \sin(x) \) with \( \tanh \)

\[
L = \frac{1}{N} \sum_i (y_i - \tilde{y}_i)^2
\]

\( y_i = \sin(x_i) \)
Regression on CNN: Nonlinear functions at the output neuron

$tanh(y_i)$
Regression on CV-QC: \( \sin(x) \)

\[
R(\phi) = \exp\left(\frac{\imath a \phi}{2}\right) = \exp\left(\frac{\phi}{2} \left( e^{\frac{\imath}{2}} + e^{-\frac{\imath}{2}} - 1 \right)\right).
\]

\[
\mathcal{S}(\psi) = \exp\left(\frac{\imath}{2} (\psi^2 - \overline{\psi}^2)\right).
\]

Details:
- Number of wires: 1
- Number of parameters: 1
- Gradient recipe: None (uses finite difference)

Parameters:
- \( \phi \) (float) — rotation angle

---

Kerr interaction.

\[
K(\kappa) = e^{\imath \kappa \phi^2}.
\]

Details:
- Number of wires: 1
- Number of parameters: 1
- Gradient recipe: None (uses finite difference)

Parameters:
- \( \kappa \) (float) — parameter

---

The position quadrature observable \( \hat{z} \).

When used with the \( \expval{\hat{z}} \) function, the position expectation value \( \expval{\hat{z}} \) is returned. This corresponds to the mean displacement in the phase space along the \( \hat{a} \) axis.

Details:
- Number of wires: 1
- Number of parameters: 0
- Observable order: 1st order in the quadrature operators

Parameters:
- wires (Sequence<int> or int) — the wire the operation acts on
Regression on CV-QC: \( \sin(x) \)

A number of hidden layers: \( L=4 \)

\[
\begin{align*}
R(\phi_{11}) & \quad S(r_{11}, \phi_{11}) & \quad R(\phi_{12}) & \quad D(a_{11}, \phi_{11}) & \quad K(\kappa_{11}) \\
R(\phi_{41}) & \quad S(r_{41}, \phi_{41}) & \quad R(\phi_{42}) & \quad D(a_{41}, \phi_{41}) & \quad K(\kappa_{41}) \\
\end{align*}
\]

|0\rangle \quad D(a_i, \phi_i)

\[
R(\phi) = \exp\left( i\phi \hat{a}^\dagger \hat{a} \right) = \exp\left( i\frac{\phi}{2} \left( \frac{\hat{x}^2 + \hat{p}^2}{\hbar} - 1 \right) \right)
\]

\[
S(x) = \exp\left( \frac{i}{2} (x^2 \hat{a}^\dagger \hat{a} - \hat{a}^\dagger \hat{a}) \right) \quad \alpha = ae^{i\phi}
\]

\[
D(a, \phi) = D(\alpha) = \exp(\alpha \hat{a}^\dagger - \alpha^* \hat{a}) = \exp\left( -i\sqrt{\frac{\hbar}{2}} (R(\alpha)\hat{p} - \text{Im}(\alpha)\hat{x}) \right)
\]

\[
K(\kappa) = e^{i\kappa \hat{x}^2}
\]

\[
L = \frac{1}{N} \sum_i (y_i - \langle X \rangle_i)^2
\]

\[
y_i = \sin(x_i)
\]
Regression on CV-QC: \(\sin(x)\)

A number of hidden layers: \(L=4\)

\[
\begin{align*}
\mathcal{D}(a_i, 0.0) & \rightarrow R(\phi_{11}) \rightarrow S(r_{11}, \phi_{11}) \rightarrow R(\phi_{12}) \rightarrow D(a_{11}, \phi_{11}) \rightarrow K(\kappa_{11}) \\
& \rightarrow R(\phi_{41}) \rightarrow S(r_{41}, \phi_{41}) \rightarrow R(\phi_{42}) \rightarrow D(a_{41}, \phi_{41}) \rightarrow K(\kappa_{41}) \\
\langle X \rangle_i & \rightarrow L = \frac{1}{N} \sum_i (y_i - \langle X \rangle_i)^2 \\
y_i & = \sin(x_i)
\end{align*}
\]
Regression on CV-QC: amplitude encoding
Regression on CV-QC: amplitude encoding

Amplitude encoding

$$\text{sigmoid}(y_i)$$
Regression on CV-QC: \( \sin(x) \)

A number of hidden layers: \( L=4 \)

\[
R(\phi_{11}) \rightarrow S(r_{11}, \phi_{11}) \rightarrow R(\phi_{12}) \rightarrow D(a_{11}, \phi_{11}) \rightarrow K(\kappa_{11}) \\
R(\phi_{41}) \rightarrow S(r_{41}, \phi_{41}) \rightarrow R(\phi_{42}) \rightarrow D(a_{41}, \phi_{41}) \rightarrow K(\kappa_{41})
\]

\[
|0\rangle \rightarrow D(a_i, \phi_i) \rightarrow \langle X \rangle_i
\]

\[
\begin{align*}
R(\phi) &= \exp(i\phi^a_i a_i) = \exp\left(i\frac{\phi}{2} \left( \frac{\hat{p}^2 + \hat{p}^2}{\hbar} - 1 \right) \right) \\
S(z) &= \exp\left(\frac{1}{2}(\hat{x}^a \hat{a}^\dagger - \hat{a}^\dagger \hat{a})\right) \\
D(a, \phi) &= D(\alpha) = \exp(\alpha^a_i a_i - \alpha^* a_i) = \exp\left(-i\sqrt{\frac{\hbar}{2}} (\text{Re}(\alpha) \hat{p} - \text{Im}(\alpha) \hat{x}) \right) \quad \alpha = a e^{i\phi} \\
K(\kappa) &= e^{i\kappa z} \\
\end{align*}
\]

\[
L = \frac{1}{N} \sum_i (y_i - \langle X \rangle_i)^2 \\
y_i = \sin(x_i)
\]
Regression on CV-QC: \( \sin(x) \)

A number of hidden layers: \( L=4 \)

DRL.de • Chart 24 > Lecture > Author • Document > Date

\[
\langle X \rangle_i = \frac{1}{N} \sum_i (y_i - \langle X \rangle_i)^2
\]

\[
y_i = \sin(x_i)
\]

\[
\alpha = ae^{i\phi} = a(\cos \phi + isin\phi)
\]

\[
S(z) = \exp\left(\frac{1}{2}(z^*a^2 - za^2)\right)
\]
Regression on CV-QC: sin(x)

A number of hidden layers: L=4

\[
\begin{align*}
\alpha &= a e^{i\phi_i} = a(\cos \phi_i + i\sin \phi_i) \\
L &= \frac{1}{N} \sum_i (y_i - \langle X \rangle_i)^2 \\
y_i &= \sin(x_i)
\end{align*}
\]
Regression on CV-QC: angle encoding
Regression on CV-QC: angle encoding

$$\tanh(y_i)$$
Conclusion

- **Regression function:** CV-QC works perfectly well (e.g., Rosenbrock function given x and y data)
- **Optimization problems:** Probably hard to use CV-QC since photons interaction is limited but we do need interactions (e.g., Rosenbrock function).
- **Classification:** CV-QC maybe not that much beneficial since **we do not need continuous values and we do have integer labels for classification tasks.** Even encoding classical datasets to quantum circuit.

\[
U(\theta)|0\rangle \approx D(a, \phi)|0\rangle
\]

DV-QC  CV-QC

- **SAR satellites:** Probably implementing CV-QC on cubesats is a very cool idea since satellites are optical computing devices.
Dream of photonic scientists

A Feynman diagram (box diagram) for photon-photon scattering: one photon scatters from the transient vacuum charge fluctuations of the other.