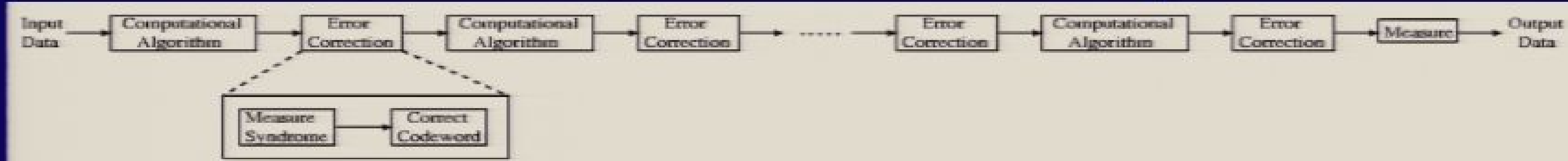
Title: Asymmetric and Adaptive Error Correction in Quantum Computation

Date: Jun 16, 2007 10:50 AM

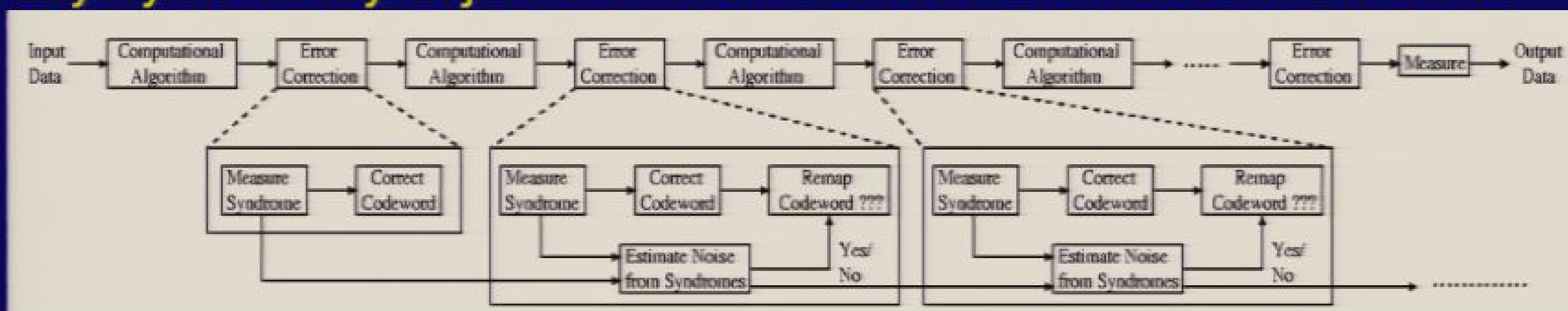
URL: http://pirsa.org/07060061

Abstract:

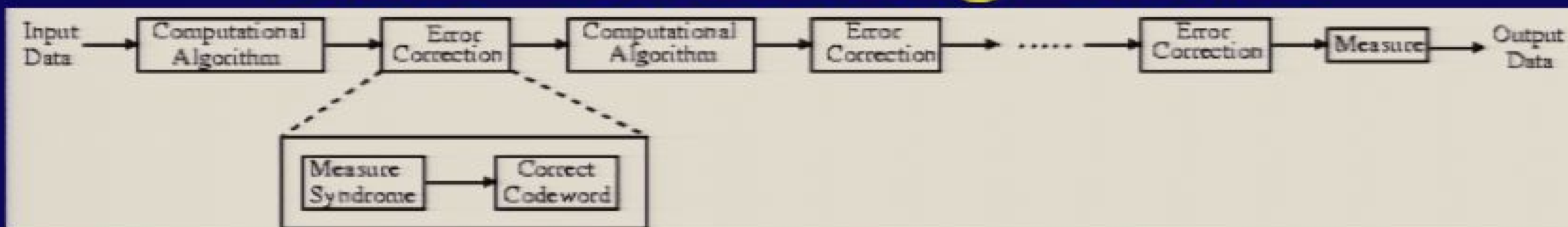
Overview – Asymmetric and Adaptive Fault Tolerant Quantum Computation



- Asymmetric Quantum Error Correction Codes: One type of noise may dominant over the other. Can we design codes for this scenario ? Yes – review existing work and set up scenario for …..
- Adaptive Error Correction: One may determine the frequency of bit flip errors and phase flip errors corrected in the error correction stage by monitoring the syndrome measurement outcomes. If the code needs a larger bit flip distance or phase flip distance, one may dynamically adjust the error correction scheme in real time.



Asymmetric Quantum Error Correcting Codes



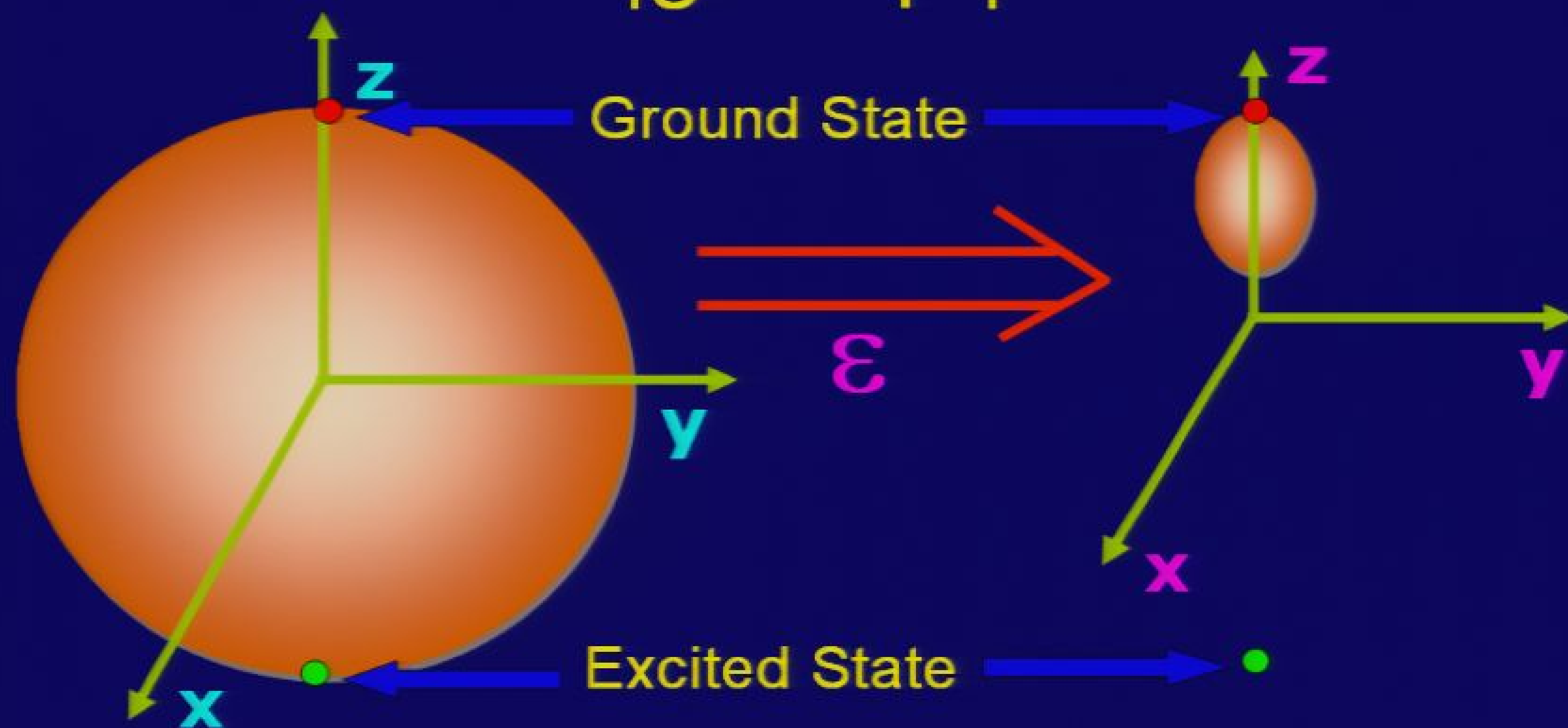
- Classical errors → Bit flips.
- Quantum errors → Bit flips & Phase flips.
- Probability of a Bit Flip = p_{bf}.
- Probability of a Phase Flip = p_{pf}.
- Quantum block code distances d_{bf} and d_{pf}.
- Efficient error correction operation.
- Asymmetric codes → $d_{bf} / d_{pf} \approx p_{bf} / p_{pf}$.

Motivational Slide: Do bf - pf error asymmetries appear in practical quantum computing architectures

- Consider the ion trap architecture.
- Ion read/write operations are dominated by laser phase noise → phase errors.
- Ions often are isolated well enough that the dominant form of decoherence is spontaneous decay from the excited state to the ground state.
- Spontaneous decay is described by the amplitude damping channel.

Amplitude Damping Dynamics

$$\Psi = \alpha\,|g\rangle + \beta\,|e\rangle$$



Unit Bloch Sphere   ε(Unit Bloch Sphere)

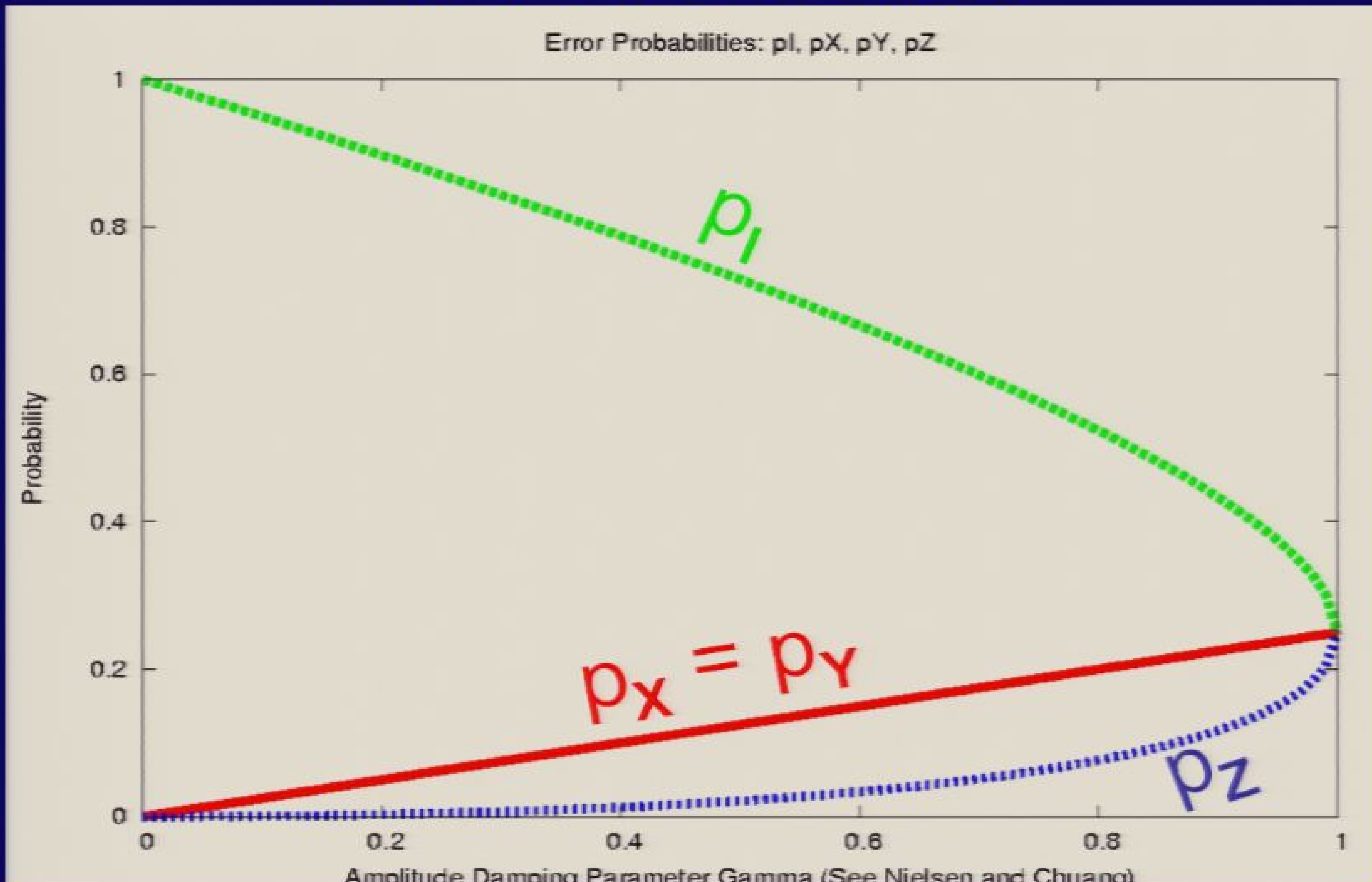
Calculating p_{bf} and p_{pf} for the amplitude damping channel

- Operator sum (Kraus) channel action

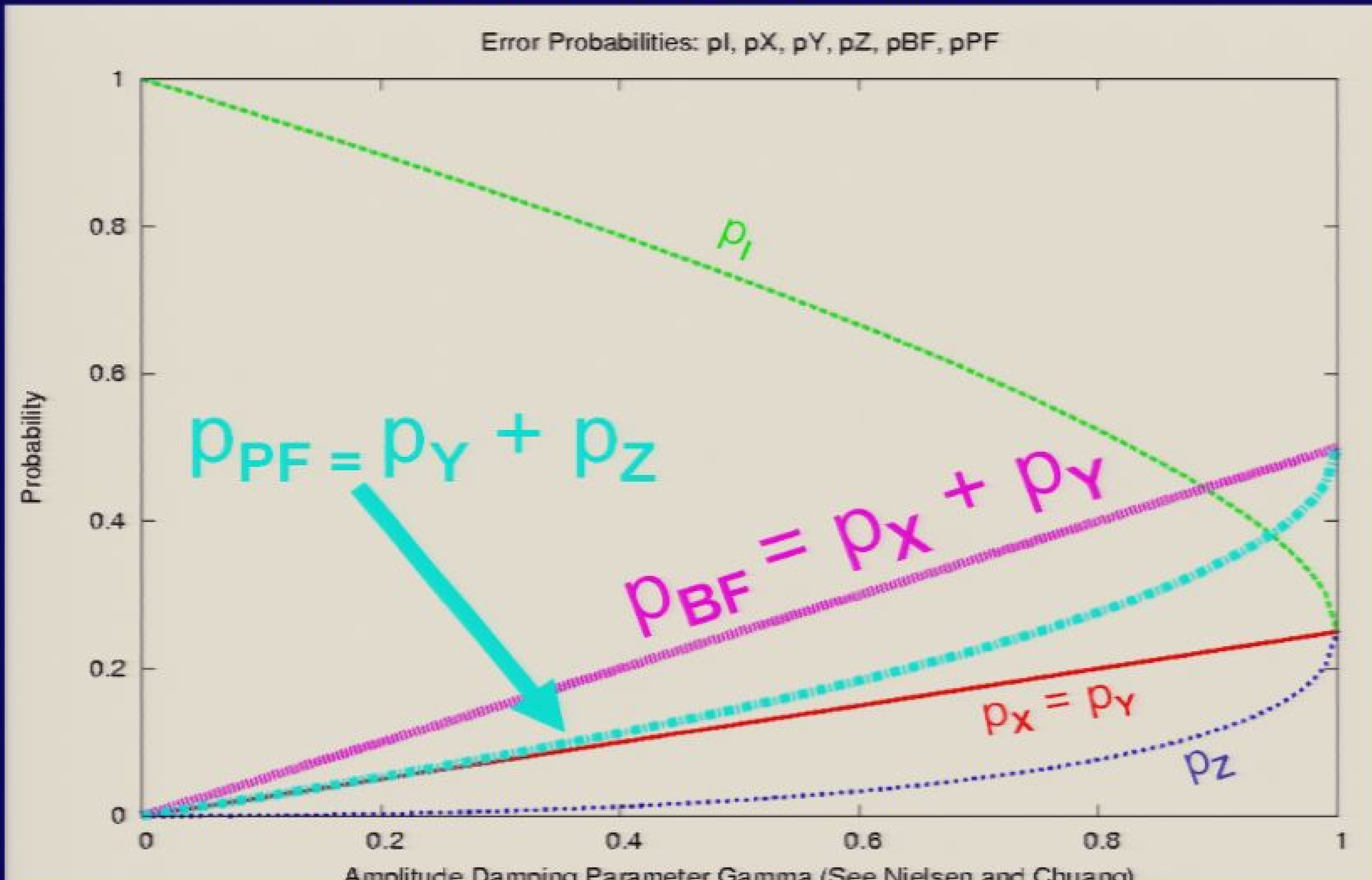
  $\mathcal{E}_{\gamma}(\rho) = \Sigma_i \; M_i \; \rho \; M^{\dagger}_i.$
- Damping parameter γ.

  $\gamma=0$ no damping, $\gamma=1$ full damping.
- Consider using the Shor nine qubit code.
- Let the damping action occur on the 1st qubit only.
- $p_{\mathbf{x}}(\rho) = \mathrm{Trace}[\, (X \otimes I^{\otimes 8})\, \rho\, (X \otimes I^{\otimes 8})\; \mathcal{E}_{\gamma}(\rho)\,]$

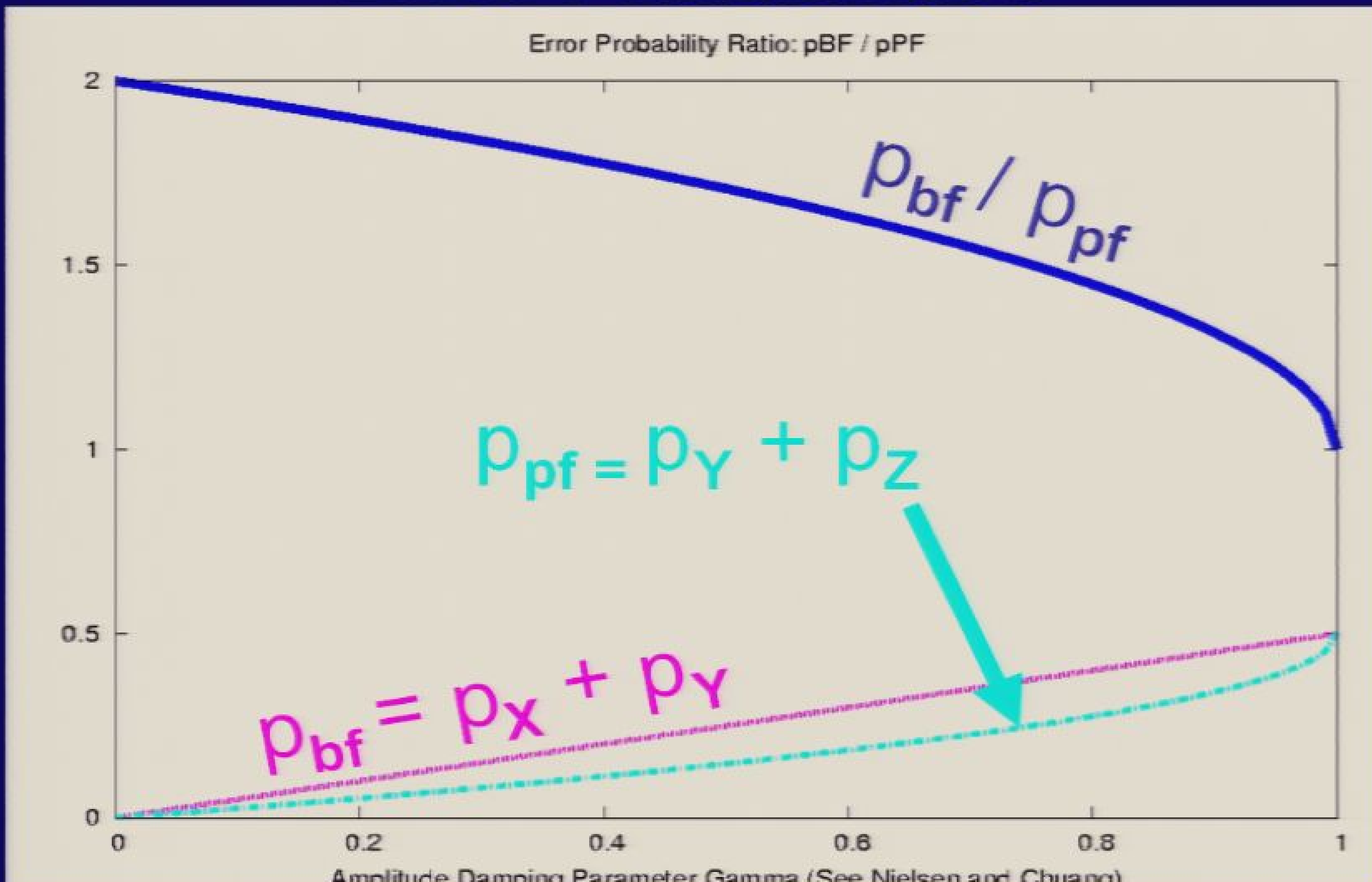
Error Probabilities



Error Probabilities



Error Probabilities



Asymmetric Quantum Codes

Examples

- Bit flip (only) code $0_L \rightarrow 000$, $1_L \rightarrow 111$.
- Bit flip distance = 1, Phase flip distance = 0.
- Similar idea yields a Phase flip (only) code.
- One may efficiently design asymmetric quantum codes with various blocklengths, encoded number of qubits and bit flip and phase flip code distances using ideas from quantum cyclic code constructions. Basic idea drawn from Calderbank et al. GF(4) paper 9608006 & Preskill, Chapter 7 notes.

Constructing Asymmetric Codes

A brief review of classic cyclic code construction

- Message polynomial m(s) of encodes k information bits $\{ i_0, i_1, i_2, \ldots, i_{k-1} \}$ as
- $m(s) = i_0 + i_1 s + i_2 s^2 + \ldots + i_{k-1} s^{k-1}$ (degree k-1)
- Code word polynomial encodes n codeword bits $\{ c_0, c_1, \ldots, c_{n-1} \}$ as
  $c(s) = c_0 + c_1 s + c_2 s^2 + \ldots + c_{k-1} s^{n-1}$ (degree n-1)
- Generator polynomial g(s) implements the encoding: $c(s) = g(s)\, m(s)$.
- The polynomial g(s) has degree n - k.
- g(s) is the key to the code properties.

Quantum cyclic code construction

- Consider two blocklength n codes, C_α and C_β.
- Use Calderbank Shor Steane code construction
- Use polynomial description of cyclic codes.
- $C_\alpha(s) = g_\alpha(s)\, m_\alpha(s)$, $C_\beta(s) = g_\beta(s)\, m_\beta(s)$.
- Desire $C_\beta \subseteq C_\alpha \rightarrow k_\alpha > k_\beta$ and $m_\beta(s) \subseteq m_\alpha(s)$.
- c(s) a linear cyclic code iff g(s) is a factor of s^n-1.
- $s^n - 1 = \Pi_j\, f_j(s)$, $j \in \Lambda$ and the $f_j(s)$ are irreducible.
- Must have $g_\alpha(s)$ divide $g_\beta(s)$.

Quantum cyclic code construction

- c(s) a linear cyclic code iff g(s) is a factor of s^n-1.
- $s^n - 1 = \Pi_j f_j(s)$, $j \in \Lambda$ and the $f_j(s)$ are irreducible.
- Desire $C_\beta \subseteq C_\alpha \rightarrow$ Must have $g_\alpha(s)$ divide $g_\beta(s)$.
- Construct $g_\beta(s) = \Pi_\lambda f_\lambda(s)$, with $\{\lambda\} \equiv \Omega \subset \Lambda$, $\Omega \neq \Lambda$.
- Construct $g_\alpha(s) = \Pi_\mu f_\mu(s)$, with $\{\mu\} \equiv \vartheta \subset \Omega$, $\vartheta \neq \Omega$.
- $d(C_\alpha) \rightarrow d_{bf}$ and $d(C^\perp{}_\beta) \rightarrow d_{pf}$.
- The irreducible factors of $s^n - 1$ used in constructing $g_\alpha(s)$ and $g_\beta(s)$ determine the quantum code properties.

Example: a class of quantum codes for fixed n with different (d_{bf}, d_{pf}).

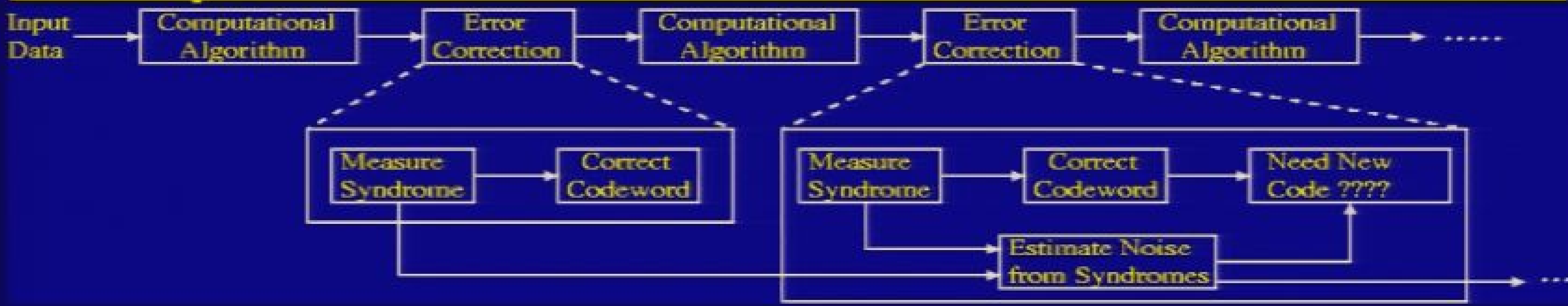
- A large # of irreducible factors for $s^n - 1$ leads to many different irreducible factor combinations and many different possible $C_\beta \subseteq C_\alpha \rightarrow g_\alpha \mid g_\beta$ constructions with different (d_{bf}, d_{pf}) pairs.
- Eg: $n=15$, $s^{15}-1 = f_1(s)\, f_2(s)\, f_3(s)\, f_4(s)\, f_5(s) = (s+1)(s^2+s+1)(s^4+s^3+s^2+s+1)(s^4+s+1)(s^4+s^3+1) \bmod 2$.
- $g_\alpha = f_3\, f_4$, $g_\beta = f_3\, f_4\, f_5 \rightarrow d(C_\alpha) = 5$ (d_{bf}), $d(C^\perp{}_\beta) = 2$ (d_{pf}).
- $g_\alpha = f_1\, f_2\, f_3$, $g_\beta = f_1\, f_2\, f_3\, f_4 \rightarrow d(C_\alpha) = 4$, $d(C^\perp{}_\beta) = 3$.

Stabilizer picture yields encoded X, Z operations

- Recall stabilizer operators $S = \{ S_X, S_Z \}$ where
  S_X = X's in parity check locations of C_α,
  S_Z = Z's in parity check locations of $C^\perp_\beta$.
- Normalizer of the Pauli group is the group of elements which commute with all the stabilizer elements. The elements of the normalizer not in the stabilizer yield the encoded X, Z operations.
- End of review on how to construct asymmetric codes and their encoded X, Z operations.

Adaptive Error Correction

Adaptive Quantum Error Correction



- Adaptation and feedback useful and common in classical signal processing.
- Use syndrome outcomes to adaptively adjust the quantum code so that adequate, but not excessive, code distances contain the errors.
- Fault tolerant gate constructions must also be updated when the code used is changed.
- By keeping the quantum code structure closely tied to the noise level present in the system, one keeps the computational resources (time and gates) needed for error correction to a minimum.

Adaptive Code Update Options

- Swap distances between bf's and pf's for fixed n.

  ⟶ Increase/decrease d_{bf} at the expense of d_{pf}.
- Change codeword length, thereby shrinking/increasing d_{bf}, d_{pf} or both.
- Keep the codeword length constant, but change the # of encoded qubits by changing the d_{bf}, d_{pf} per qubit ?

(Where do you store the swapped out encoded qubits ?)
- Combinations of the above.
- These degrees of freedom exist in quantum code design and can be understood and manipulated using the asymmetric techniques previously outlined.

When is adaptation useful ?

- Adaptation works best for slowly varying noise.

Gate time scale << noise variation time scale

and

Noise variation time scale << total execution time.

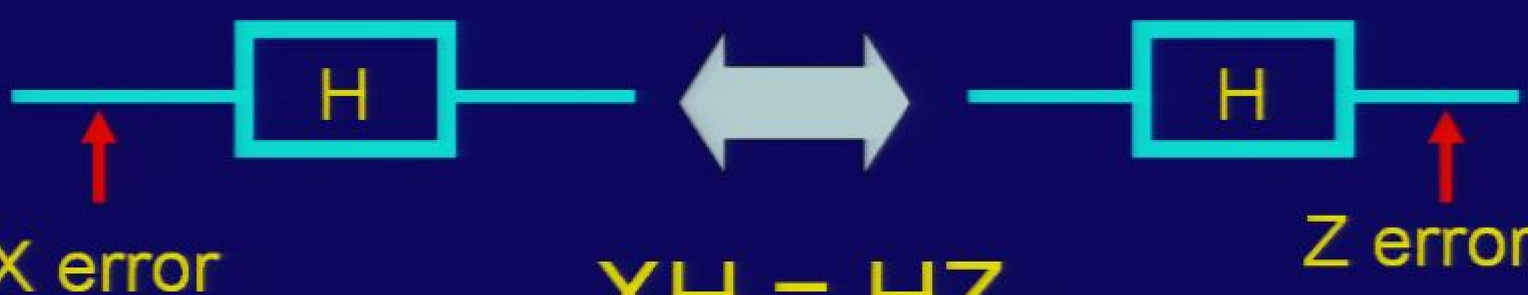
- In this case, one does not change the underlying code structure too often, but often enough to aid overall execution performance.
- One may also wish to utilize different codes for different computational stages/operations: moving qubits, laser read/write, gate operations.

Computational Gate Error Analysis

- The determination of the relative probability of bf versus pf errors is complicated by error conversion due to computational gates.
- Appropriate bit flip, phase flip code distances for upcoming computational blocks depend on the anticipated gates to be implemented, as well as consideration of the computational gates executed in previous computational stages from which statistics were extracted.
- → Make broad assumptions to reduce the complexity of estimating the error probabilities.

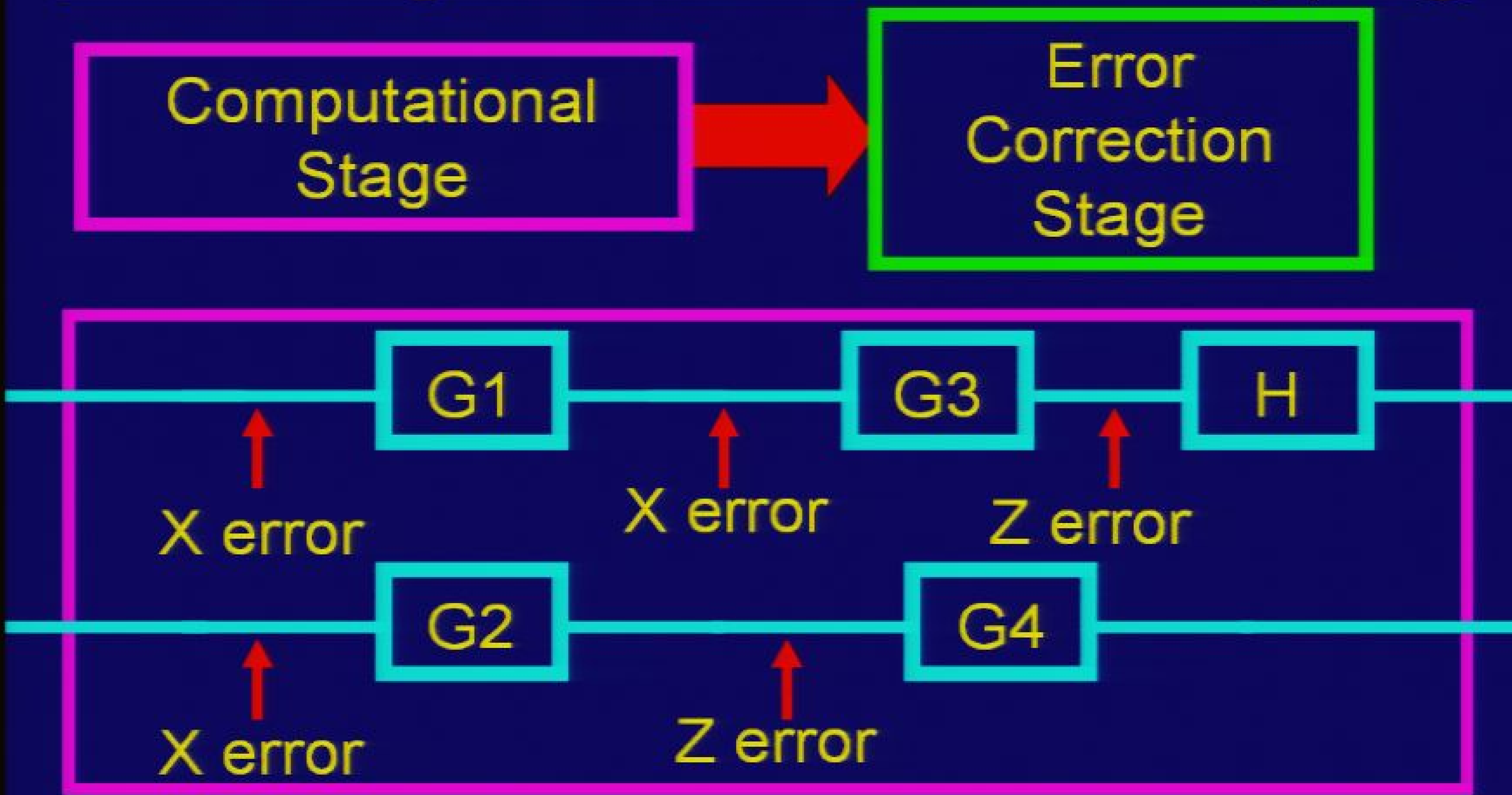
Computational Gate Error Conversion

- Computational gates may convert one type of error into another.
- Example: bit flip error → phase flip error
- Consider the Hadamard gate H. Recall XH = HZ. A bit flip error before H propagates as a phase flip error after the H gate. Depending on whether the bit flip error occurs before or after the Hadamard affects which error propagates.



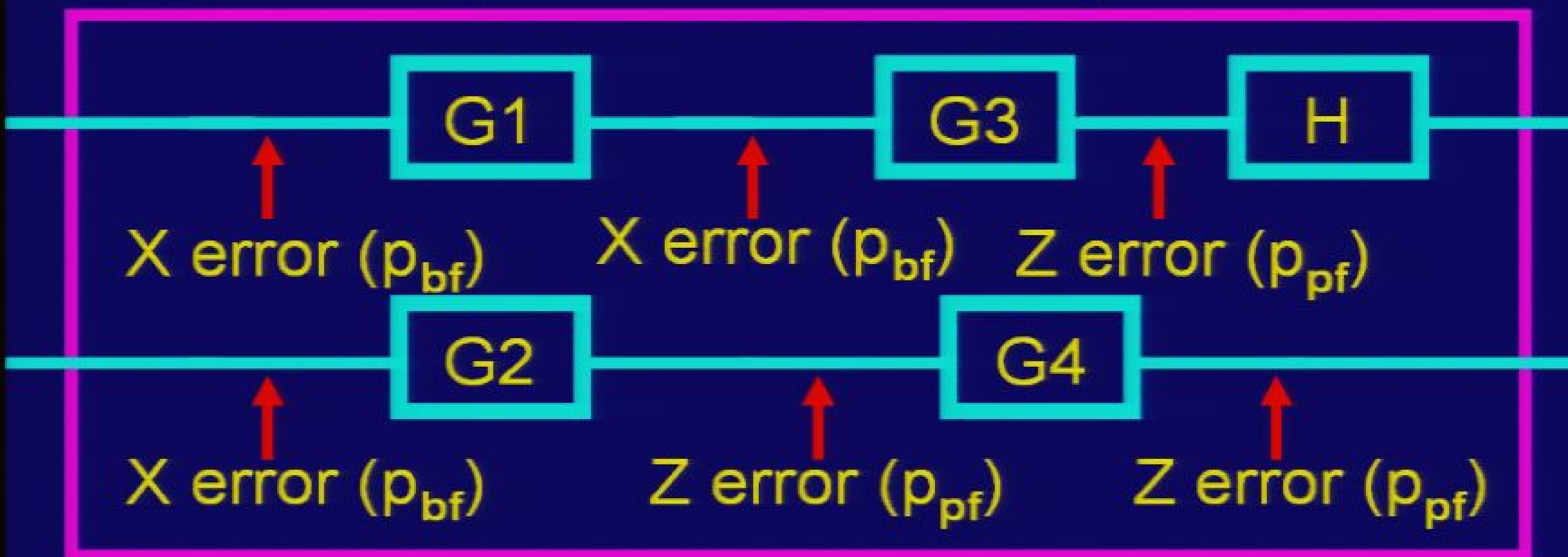
$$XH = HZ$$

Estimating the Errors & Bootstrapping



How is one to track all these error probabilities ?

Error Estimation

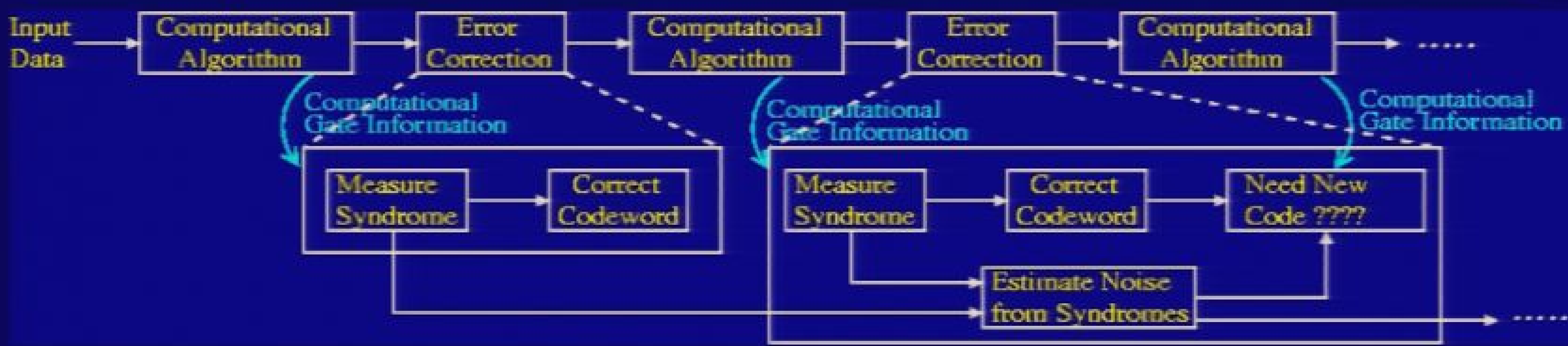


- Make simplifying error assumptions.
- Use the observed syndrome distribution to predict how the noise parameterization degrees of freedom are behaving.
- Use error parameterization + gate information to predict code properties for upcoming computational blocks.

Adaptive Algorithm Summary

- Precompile in an ancillary classical computer a range of [n, k, d_{bf}, d_{pf}] quantum codes with their corresponding fault tolerant implementations.
- Make generalizing assumptions about noise behavior and determine what the error correction syndromes indicate about the noise parameterization degrees of freedom.
- Analyze syndrome measurement outcomes, in light of past and upcoming computational gates to estimate code requirements for future blocks.
- Implement a new code and corresponding fault tolerant gates as needed.

Is adaptive error correction worth the trouble ?



- Adaptation tracks the error picture, minimizing
  - ✓ Codeword lengths
  - ✓ # of syndromes to be measured
  - ✓ # of qubits per syndrome measurement

Code	n	k	d_{bf}	d_{pf}	# of Syndromes	# of Qubits per Syndrome
Bit Flip	3	1	3	0	2	2
Steane	7	1	3	3	6	4

Conclusion & Summary

- Adaptive error schemes yield improved efficiency in quantum communication and quantum computation architectures which have time varying bit flip/phase flip noise levels. Time varying noise may occur due to quantum computer/communication hardware modifications and/or as equipment ages.
- Approach may be useful in simplifying detailed error correction architecture design. (Error noise locking.)
- Asymmetric and adaptive error correction schemes maximize the time spent by the quantum computer executing the algorithm of interest, while minimizing the resources and time needed to implement error correction.