# **Roadmap for Advancing H2MM Pipeline:**

**I. Enhancing Model Fidelity & Objectivity (Inspired by BNP-FRET & Best Practices):**

1. **Objective Determination of Nstates (High Priority):**
   * **Current:** Pre-specified Nstates.
   * **Improvement:**
     + **Implement BIC/AIC:** Loop runHMM\_local.m (with your chosen InitialGuess\_\*.m function) for a range of Nstates. Calculate BIC and AIC. Select Nstates that minimizes these criteria. This is the most practical first step.
     + **(Advanced/Long-term):** Explore Variational Bayes HMMs (VB-HMMs) or investigate implementing aspects of Bayesian Nonparametric approaches (like those in BNP-FRET, e.g., using a Dirichlet Process prior over transition probabilities within an iHMM framework if you were to build a new sampler). This is a significant research direction.
   * **GPU Impact:** Faster training per Nstates makes exploring a wider range feasible.
2. **Explicit Modeling of Photophysics (Medium Priority, depending on data):**
   * **Current:** Photophysics (blinking, bleaching) is implicitly handled or might require specific hardcoding in h2mm\_main\_blink.m.
   * **Improvement (Inspired by H2MM & BNP-FRET):**
     + **Add Dark/Blinked States:** Introduce explicit states for fluorophore dark states (e.g., triplet states, radical ion states leading to blinking, or irreversibly bleached states).
       - Initialize obsmat for these states with very low/zero D and A emission.
       - Initialize transmat with plausible rates for entering/exiting these dark states (e.g., blinking rates, bleaching rates).
       - Use BIC/AIC to justify the inclusion of these additional states.
     + **Parameterize Photophysical Rates:** Instead of just having a dark state, try to *learn* the rates of blinking and bleaching if they are significant and variable. This increases model complexity.
     + **BNP-FRET Approach:** The Saurabh et al. paper explicitly aims to model fluorophore photophysics as part of its "superstates" (system state + photophysical state). Adapting this concept, even within an MLE framework, would mean defining a more complex state space.
   * **GPU Impact:** More states mean larger matrices; GPU acceleration becomes even more beneficial.
3. **Incorporating Detector Effects & Background Systematically (Medium/High Priority):**
   * **Current:** Assumed to be handled by pre-processing or implicitly.
   * **Improvement (Inspired by BNP-FRET):**
     + **Crosstalk:** While your photon-by-photon assignment to D/A channels is a good start, significant crosstalk can still affect P(A|State). BNP-FRET models crosstalk explicitly. For your HMM, you could:
       - Estimate crosstalk parameters (D->A leakage, A->D leakage) independently.
       - Modify the observation likelihood P(observed\_symbol | State\_i) to account for this. E.g., P(observed\_A\_photon | State\_i) = P(true\_A\_emission | State\_i)\*(1-leak\_AtoD) + P(true\_D\_emission | State\_i)\*leak\_DtoA. This makes obsmat generation more complex.
     + **Background:** Model background as an additional source of photons (Poisson process for each channel) and incorporate this into the likelihood calculation. This is non-trivial for photon-by-photon HMMs but is standard in some binned HMM approaches. BNP-FRET explicitly addresses this.
     + **Detector Dead Time & IRF:** For very fast kinetics or pulsed experiments, these become critical. BNP-FRET discusses incorporating them. For your current continuous-wave photon-by-photon HMM, dead time might manifest as slightly altered inter-photon times. IRF is more relevant for lifetime measurements in pulsed setups.
   * **GPU Impact:** More complex likelihood calculations per photon could benefit from GPU if done in batches.

**II. Improving HMM Training & Robustness:**

1. **Advanced Initial Guess Strategies (Medium Priority):**
   * **Current:** Good variety with InitialGuess\_PhotonByPhoton.m, \_special.m, \_chainBlock.m.
   * **Improvement:**
     + **Data-Driven obsmat Initialization:** Implement kmeans clustering on binned FRET data (from a quick pre-calculation) to get initial centers for P(A|State) in obsmat0.
     + **Smarter transmat0:** Beyond diagonal dominance, explore initializations that reflect expected timescales or chain-like/cyclic models if appropriate for your system.
     + **Perturbation from Good Models:** If you have a good model from a similar dataset, use its parameters as a starting point and add small random perturbations to generate new initial guesses.
   * **GPU Impact:** Not directly, but faster training allows for more initial guesses to be explored.
2. **Ensuring Global Maximum Likelihood / Posterior (Ongoing Concern):**
   * **Current:** Multiple initial guesses help avoid some local optima in MLE.
   * **Improvement:**
     + **Simulated Annealing or Deterministic Annealing EM:** Modifications to the EM algorithm that can better escape local optima, though they can be slower.
     + **(BNP-FRET context):** MCMC methods used in Bayesian inference are designed to explore the full posterior, which inherently handles multimodality better than simple MLE optimization, but convergence needs careful assessment.
   * **GPU Impact:** Faster iterations make more sophisticated optimization/sampling schemes more feasible.

**III. Enhancing Output Analysis & Interpretation:**

1. **Comprehensive Kinetic Parameter Extraction (High Priority):**
   * **Current:** Transition counts (1->2, 2->1) from validateViterbiStates\_local.m.
   * **Improvement:**
     + **Dwell Time Analysis:** From Q0, extract dwell times for each state in each trajectory. Plot dwell time histograms. Fit to exponential (or multi-exponential) distributions to get empirical rates. Compare these with rates derived from the HMM transmat (e.g., k\_ij = transmat(i,j) / dt\_model\_sec for off-diagonals, if dt is small and self-transition is high, or from the K matrix if calculated via logm).
     + **Transition Path Analysis (for Nstates > 2):** If you identify more than two states, analyze preferred pathways and intermediate states.
     + **Uncertainty Quantification for Rates:** If using MLE, estimate uncertainties on transmat elements (e.g., via bootstrapping Viterbi paths or Fisher Information Matrix if calculable). If moving towards Bayesian HMMs, you'd get posterior distributions for rates directly.
   * **GPU Impact:** Dwell time extraction is CPU-bound. Fitting many distributions could be parallelized.
2. **Advanced Validation & Model Comparison (Medium Priority):**
   * **Current:** validateViterbiStates\_local.m FRET histograms, visual inspection from Traj\_Vet\_local.m.
   * **Improvement:**
     + **Cross-Validation:** Split data into training/testing sets. Train model on training set, evaluate log-likelihood on the (unseen) test set. This helps detect overfitting.
     + **Simulate from Fitted Model:** Simulate trajectories using your fitted HMM parameters. Compare statistics (FRET histograms, dwell times, transition counts) of simulated data to experimental data. Your GenSimRecolored\_hist in processHMM\_results\_local.m already does a form of this for FRET histograms.
     + **Residual Analysis:** For photon-by-photon, this is less standard than in time series with continuous values, but one could look for systematic deviations.
   * **GPU Impact:** Simulating many trajectories from a model can be accelerated if the simulation engine is parallelizable.

**IV. Pushing Beyond Current Literature (More Research-Oriented):**

1. **Learning Continuous State Spaces / Energy Landscapes (Very Advanced):**
   * **Inspiration:** BNP-FRET mentions the goal of analyzing systems "better modeled as continuous, such as intrinsically disordered proteins."
   * **Challenge:** Standard HMMs assume discrete states. Moving to continuous hidden states (e.g., a 1D reaction coordinate for FRET) requires different mathematical frameworks (e.g., state-space models with non-linear dynamics, or methods inferring an underlying potential energy surface).
   * **Potential:** Could reveal ruggedness or diffusional motion along a conformational landscape rather than just discrete jumps.
   * **This is a significant departure from HMMs and a frontier research topic.**
2. **Hierarchical Models / Population Heterogeneity:**
   * **Inspiration:** What if different molecules in your sample genuinely follow slightly different kinetic HMMs?
   * **Improvement:** Hierarchical Bayesian models could try to infer a distribution of HMM parameters across the population of observed trajectories, identifying sub-populations with distinct dynamics. HDP-HMMs (Hierarchical Dirichlet Process HMMs) are a step in this direction for state discovery but can be extended for parameter heterogeneity.
   * **Significance:** Moves beyond assuming all molecules are statistically identical.
3. **Physics-Informed Neural Networks (PINNs) / Hybrid DL-HMM:**
   * **Idea:** Train a neural network to perform parts of the analysis (e.g., denoising, feature extraction from photon patterns, or even estimating parts of the generator matrix G from the BNP-FRET paper) but constrain its learning with known physical laws (e.g., detailed balance for transitions, photophysical rules).
   * Could combine the feature learning power of DL with the interpretability/constraints of physical models.
   * **This is cutting-edge and research-intensive.**
4. **Real-time Adaptive HMM Analysis:**
   * For experiments where data is acquired over long times or conditions change, could the HMM parameters be updated adaptively or in real-time? This is very challenging but could be powerful for tracking non-stationary systems.

**GPU Acceleration Strategy (General):**

* **Identify Bottlenecks:** Always start by profiling your existing code (h2mm\_main\_blink.m, fwdback\_photonByphoton\_fast, CalculatePowerOfTransMatrices).
* **MATLAB's GPU Capabilities:** Leverage gpuArray and GPU-enabled built-in functions for linear algebra and array operations first. This is the easiest entry point.
* **Custom Kernels (If Necessary):** For highly specific, performance-critical loops that don't map well to built-ins, consider writing custom CUDA kernels (requires C++/CUDA knowledge).
* **Batch Processing:** GPUs love large, independent batches of work. Structure your data and computations accordingly (e.g., process multiple trajectories or multiple inter-photon durations in parallel on the GPU).
* **Data Transfer Management:** Minimize CPU-GPU data transfers. Perform as many sequential operations as possible on the GPU before bringing results back to the CPU.

**Summary for Moving Forward:**

1. **Solidify the Foundation:**
   * Implement **BIC/AIC for Nstates selection**.
   * Refine **initial guess strategies** (e.g., kmeans-guided obsmat).
   * Ensure your **time unit handling** is flawless throughout all scripts.
   * Thoroughly **validate the GPU acceleration** of your current h2mm\_main\_blink.m if you proceed with that.
2. **Incorporate More Physics (Iteratively):**
   * Start with **explicit dark/blinking states** if relevant. Use BIC/AIC to justify.
   * Consider how to model/correct for **crosstalk and background** more explicitly if they are significant issues.
3. **Enhance Output Analysis:**
   * Implement robust **dwell time analysis** and compare empirical rates with HMM transmat-derived rates.

By systematically tackling these, we will significantly enhance our current H2MM pipeline, borrowing strengths from advanced concepts while keeping the implementation manageable. The "beyond current literature" steps are more like research projects that could spin off from a very solid, well-validated core pipeline.

Okay, let's break down AIC and BIC, both generally and specifically for your HMM Nstates scan results.

**Information Criteria: The General Idea**

When we build statistical models, we often face a trade-off:

1. **Goodness of Fit:** We want a model that explains the observed data well. More complex models (with more parameters) can usually achieve a better fit to the *training data*.
2. **Parsimony (Simplicity/Generalizability):** We also want a model that is not *too* complex. Overly complex models might fit the noise in the training data perfectly but generalize poorly to new, unseen data (this is called overfitting). Simpler models are often more robust and easier to interpret.

**Information Criteria (like AIC and BIC) are statistical measures that help us select among a set of candidate models. They try to balance goodness of fit with model complexity.**

* They all start with a term related to the **maximized log-likelihood (LL)** of the model. A higher LL means a better fit to the data.
* They then add a **penalty term** that increases with the number of parameters in the model.

The model with the **lowest** AIC or BIC value is generally preferred.

**AIC (Akaike Information Criterion)**

* **Formula (common form):** AIC = -2 \* LL + 2 \* k
  + LL: The maximized log-likelihood of the model.
  + k: The number of estimable parameters in the model.
* **Interpretation:**
  + AIC estimates the prediction error and thereby the relative quality of statistical models for a given set of data.
  + It's founded in information theory. It tries to select the model that minimizes the (estimated) information lost when using the model to represent the process that generates the data.
  + The 2\*k term is the penalty for adding parameters.
* **Properties:**
  + AIC is not a test of a null hypothesis (i.e., it doesn't tell you if a model is "true" or "false"). It's a tool for *model selection* among a candidate set.
  + It tends to favor more complex models than BIC, especially when the number of data points (N) is large. This is because its penalty term (2k) does not depend on N.
  + It's considered "asymptotically efficient" if the true model is very complex (infinite dimensional). If the true model is simple and in the candidate set, AIC has a tendency to pick a more complex model than necessary.

**BIC (Bayesian Information Criterion or Schwarz Criterion)**

* **Formula (common form):** BIC = -2 \* LL + k \* log(N)
  + LL: The maximized log-likelihood of the model.
  + k: The number of estimable parameters in the model.
  + N: The number of data points (observations) used to fit the model.
* **Interpretation:**
  + BIC is derived from a Bayesian argument. It approximates the Bayes factor (which compares the posterior probability of one model versus another).
  + The k \* log(N) term is the penalty. Notice that this penalty depends on the sample size N. As N gets larger, the penalty for adding parameters becomes much stronger compared to AIC.
* **Properties:**
  + BIC is "consistent." This means that if the true model is among the set of candidate models, BIC will select it with a probability approaching 1 as the number of data points N goes to infinity.
  + Because of its stronger penalty (especially for large N), BIC tends to select simpler models (fewer parameters) compared to AIC.
  + It's often preferred when the goal is to identify the "true" underlying model, assuming such a model exists and is relatively simple.

**AIC vs. BIC: Which to Use?**

* **No Universal "Best":** The choice often depends on the goal.
  + If the goal is **prediction accuracy** and you believe the underlying process might be very complex and not perfectly captured by any of your simple models, **AIC** might be preferred as it's less likely to underfit.
  + If the goal is to identify the **most parsimonious "true" model** (assuming one exists within your candidates) and you want to avoid including unnecessary parameters, **BIC** is often preferred due to its consistency and stronger penalty for complexity.
* **Sample Size:** For small N, BIC's penalty k\*log(N) might be smaller than AIC's 2k, potentially leading BIC to select more complex models than AIC (though this is less common in practice with typical datasets). For large N, BIC's penalty is much larger.
* **Common Practice:** It's good practice to report both AIC and BIC. If they agree, it strengthens your model choice. If they disagree, you need to think carefully about the implications and potentially use other validation methods or domain knowledge.

**In the Context of Your HMM Nstates Scan Data Analysis:**

Let's re-examine your latest results table (with 25k iterations):

Nstates Best\_LL Num\_Params\_k AIC BIC Mean\_Duration\_Per\_Guess\_sec

\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2 -3408 5 6826.1 6858.8 19.515

3 -3388 11 6798.1 6870 60.166

4 -3376.2 19 6790.5 6914.7 275.18

5 -3375 29 6808.1 6997.8 389.84

6 -3371.9 41 6825.7 7093.8 904.28

Here, N (number of data points for BIC) is N\_photons\_total (which was 5115 in your log). log(5115) ≈ 8.54.

* **k (Num\_Params\_k):**
  + N=2: k = (2-1)prior + 2\*(2-1)trans + 2\*(2-1)obs = 1+2+2 = 5 (Correct)
  + N=3: k = (3-1) + 3\*(3-1) + 3\*(2-1) = 2+6+3 = 11 (Correct)
  + N=4: k = (4-1) + 4\*(4-1) + 4\*(2-1) = 3+12+4 = 19 (Correct)
  + N=5: k = (5-1) + 5\*(5-1) + 5\*(2-1) = 4+20+5 = 29 (Correct)
  + N=6: k = (6-1) + 6\*(6-1) + 6\*(2-1) = 5+30+6 = 41 (Correct)  
    Your calculation of k is correct (assuming 2 observation symbols and obs\_learn\_flag\_for\_k = 1).
* **Interpretation of Your Results:**
  + **Best\_LL:** Steadily increases as Nstates increases. This is expected – a more complex model can always achieve at least as good a fit (and usually better) to the training data. The improvement in LL becomes smaller after N=4.
  + **AIC:**
    - AIC = -2\*LL + 2\*k
    - The penalty for adding states (increasing k) is relatively small (2\*k).
    - AIC decreases from N=2 to N=3 to N=4, indicating that the improvement in LL is outweighing the 2\*k penalty.
    - At N=5, AIC starts to increase. This means the improvement in LL from N=4 to N=5 (-3376.2 to -3375.0, gain of 1.2 LL units) is *less than* the additional penalty from the 10 extra parameters (2 \* (29-19) = 20). 2 \* 1.2 = 2.4 < 20, so AIC increases.
    - **AIC selects Nstates = 4.** It suggests that a 4-state model offers the best trade-off between fit and complexity from its perspective.
  + **BIC:**
    - BIC = -2\*LL + k\*log(N\_photons\_total)
    - Here, log(N\_photons\_total) = log(5115) ≈ 8.54. The penalty per parameter is ~8.54\*k, which is much larger than AIC's 2\*k.
    - For N=2 to N=3:
      * LL improves by 2\*(-3388 - (-3408)) = 2\*20 = 40.
      * Penalty increases by (11-5)\*log(5115) = 6 \* 8.54 ≈ 51.24.
      * Since 40 < 51.24, BIC increases (6858.8 to 6870).
    - **BIC selects Nstates = 2.** It heavily penalizes the 6 extra parameters needed for N=3 because the LL gain doesn't compensate for this larger penalty.

**Your Context: Adenylate Kinase (AK) - Biologically 2 Main States**

* **BIC aligns with your biological expectation.** This is common. If there's a relatively simple "true" underlying process, BIC is good at finding it.
* **Why does AIC pick N=4?**
  + **Real Sub-states/Intermediates:** AK, while having two *major* conformations (open and closed), is known to potentially have short-lived intermediate states during its catalytic cycle or conformational transition (e.g., a partially closed state, ligand-bound sub-states). AIC might be sensitive enough to pick up on these if they leave a sufficient statistical signature in your photon data.
  + **Photophysics:** As discussed before, if your dyes exhibit blinking or other transient dark states, the HMM might try to model "Open-Normal, Open-Dark, Closed-Normal, Closed-Dark" effectively giving you 4 apparent FRET states. This is a very common reason for HMMs to suggest more states than expected from pure conformational changes.
  + **Heterogeneity:** Small sub-populations of AK behaving differently.
  + **Overfitting to Noise (less likely if AIC *peaks* and then increases):** The fact that AIC increases after N=4 suggests it's not just endlessly overfitting. N=4 is a genuine minimum for AIC.

**What to Do Now (Decision Making):**

1. **Trust your biological prior + BIC for the primary model:** Given your expectation of 2 main states for AK, the N=2 model (BIC winner) should be your primary focus for interpreting the main open/closed dynamics.
2. **Investigate the N=4 model (AIC winner) as an exploratory model:**
   * Carefully examine the obsmat0 (emission probabilities) and transmat0 (transition rates) of the best N=4 model.
   * Are the 4 states distinct in their FRET efficiencies?
   * **Scenario 1: Photophysics?** Do two states have high FRET (e.g., closed) but different kinetics, and two states have low FRET (e.g., open) but different kinetics? Or, more clearly, do you see two primary FRET levels, and for each, a corresponding state with very low emission probability for both donor and acceptor (a "dark" state)? If so, the N=4 model might be capturing blinking of one or both dyes in conjunction with the two conformational states.
   * **Scenario 2: Intermediates?** Are there, for example, two high-FRET states, one mid-FRET, and one low-FRET state, all with significant populations and interpretable transitions? This could point to a real intermediate.
   * **Scenario 3: Redundant states?** Are two of the four states very similar in both emission and transition probabilities to another pair? This might mean N=4 is trying to split what is essentially one broader state.
3. **Compare LL values:** The jump in LL from N=2 to N=3 (~20) and N=3 to N=4 (~12) is notable. The jump from N=4 to N=5 (~1.2) is very small. This "elbow" also suggests that N=4 captures most of the explainable variance, and beyond that, you're adding parameters for very little gain in fit.
4. **Consider the Mean\_Duration\_Per\_Guess\_sec:** While not directly a model selection criterion, significantly longer times for convergence (if not just due to more parameters but due to a "flatter" likelihood landscape) can sometimes hint that the model is less well-defined by the data. However, N=4 is not excessively long compared to N=5/6 in your case.

**Final Recommendation for Your AK System:**

* **Report N=2 as your primary model**, justified by BIC and strong biological expectation. Analyze its parameters (FRET efficiencies, rates) in detail.
* **Discuss N=4 as the AIC-preferred model.** Present its parameters and try to interpret what the additional states might represent (photophysics is a strong candidate if you haven't explicitly modeled it). If the N=4 model clearly separates into, say, two conformational states each with a linked dark/blinking state, then AIC has helped you identify this extra complexity. If the extra states are not clearly interpretable, then N=2 remains the more robust choice.
* The fact that AIC starts increasing after N=4 gives you confidence that you are not grossly underfitting by stopping your consideration at N=4.

This combined approach (using both criteria and then inspecting the models) is standard practice.

You're absolutely right about transmat relating to transition rates. Let's clarify the physical meaning of all three core HMM parameters in the context of your smFRET experiments:

1. **prior (Initial State Probability Vector):**
   * **Mathematical Definition:** prior(i) is the probability that the molecule is in hidden state i at the **very beginning of the observation period for each trajectory** (i.e., at the time of the first photon in each burst, or at t=0 if you were modeling from the absolute start of data acquisition).
   * **Physical Meaning in Your smFRET Context:**
     + When a new molecule (burst) enters the laser spot and emits its first photon, prior(i) represents the probability that this molecule was in conformational state i (e.g., "Open" or "Closed" for AK) just as it started being observed.
     + **If you assume your system reaches a steady state (equilibrium) between bursts or before observation starts:** The prior vector should ideally reflect this steady-state distribution of conformational states. This is why in processHMM\_results\_local.m, we calculate a Prior\_steady\_state\_calc from the Trans matrix and often use that as the "true" prior for interpretation, even though the HMM learns its own prior from the start of each burst.
     + **If observation starts far from equilibrium (e.g., immediately after a perturbation):** The learned prior might reflect a non-equilibrium starting distribution.
   * **Dimensions:** It's a row vector of size [1, Nstates], and sum(prior) must equal 1.
2. **transmat (Transition Probability Matrix):**
   * **Mathematical Definition:** transmat(i,j) is the probability that if the molecule is in hidden state i at a given time step, it will transition to hidden state j in the *next discrete time step*.
   * **Physical Meaning in Your smFRET Context (Photon-by-Photon H²MM):**
     + This is where H²MM (and your Factorization approach) is more nuanced than a simple discrete-time HMM. Your transmat\_init (and the transmat learned by h2mm\_main.m) is the transition probability matrix for a **small, fundamental unit time step, TAU** (which you've set to 1, likely corresponding to dt\_instrument\_ns or 1 ns).
     + transmat(i,j) for i != j is related to the **rate of transition** from conformational state i to conformational state j. Specifically, if k\_ij is the rate constant (units of 1/time) for i -> j, then for a very small TAU, transmat(i,j) ≈ k\_ij \* TAU.
     + transmat(i,i) is the probability of *staying* in state i during that small TAU, so transmat(i,i) ≈ 1 - (sum of k\_ik for k!=i) \* TAU.
     + **The K matrix** you calculate in processHMM\_results\_local.m (K = (Trans - eye(Nstates)) / dt, where Trans is the sorted, learned transmat and dt is dt\_analysis\_sec) is an attempt to extract the **continuous-time rate matrix**. The off-diagonal elements of K (K(i,j) for i != j) are direct estimates of the rate constants k\_ij (in s⁻¹ if dt is in seconds).
     + The h2mm\_main.m function, using TotalArrivalDelta and R = Factorization(...), effectively calculates transmat\_t = transmat ^ delta\_t\_photons to find the transition probability over the actual (variable) time interval between photons.
   * **Dimensions:** It's an [Nstates, Nstates] matrix, and each row sum(transmat(i,:)) must equal 1.
3. **obsmat (Observation or Emission Probability Matrix):**
   * **Mathematical Definition:** obsmat(i,o) is the probability of observing symbol o when the molecule is in hidden state i.
   * **Physical Meaning in Your smFRET Context:**
     + **Hidden States (i):** These are your Nstates conformational states (e.g., AK-Open, AK-Closed, or potentially AK-Open-Dye1Dark, etc., if you model photophysics).
     + **Observation Symbols (o):** In your typical 2-color smFRET setup, you have two observation symbols:
       - o=1: Photon detected in the Donor channel.
       - o=2: Photon detected in the Acceptor channel.
     + So, obsmat(i, 1) is the probability that if the molecule is in conformational state i, the next detected photon (from that molecule, ignoring background for a moment) will be a Donor photon.
     + obsmat(i, 2) is the probability that if the molecule is in conformational state i, the next detected photon will be an Acceptor photon.
     + **Relation to FRET Efficiency:** For a given state i, its characteristic FRET efficiency E\_i is directly related to these emission probabilities. If we ignore different quantum yields of dyes and detection efficiencies for a moment, then:
       - E\_i ≈ obsmat(i, 2) (Probability of acceptor emission)
       - And obsmat(i, 1) ≈ 1 - E\_i (Probability of donor emission)
       - More accurately, E\_i = obsmat(i,2) / (obsmat(i,1) + obsmat(i,2)) if obsmat rows represent *relative emission rates* before normalization. Since your obsmat rows are normalized to sum to 1 (i.e., obsmat(i,1) + obsmat(i,2) = 1), then obsmat(i,2) directly represents the FRET efficiency (or more precisely, the probability of an acceptor channel photon given the system is in state i and emits a photon).
     + When you sort your states by Obs\_sort = obsmat\_chosen(:,2)' in processHMM\_results\_local.m, you are sorting the states based on their FRET efficiency (probability of acceptor emission).
   * **Dimensions:** It's an [Nstates, NumberOfObservationSymbols] matrix. In your 2-color case, it's [Nstates, 2]. Each row sum(obsmat(i,:)) must equal 1.

**In Summary for Your AK Experiment:**

* **prior:** The fraction of AK molecules that are in the "Open" state vs. "Closed" state (and any other identified states) at the moment they first enter the laser beam and emit a photon.
* **transmat (and the derived K matrix):**
  + K(Open, Closed): The rate constant (e.g., in s⁻¹) at which an open AK molecule transitions to the closed conformation.
  + K(Closed, Open): The rate constant at which a closed AK molecule transitions to the open conformation.
  + Diagonal elements of K are negative and their magnitude is the sum of outgoing rates from that state.
* **obsmat:**
  + obsmat(Open, DonorChannel): Probability of a donor photon if AK is Open.
  + obsmat(Open, AcceptorChannel): Probability of an acceptor photon if AK is Open (this is effectively the FRET efficiency of the Open state).
  + obsmat(Closed, DonorChannel): Probability of a donor photon if AK is Closed.
  + obsmat(Closed, AcceptorChannel): Probability of an acceptor photon if AK is Closed (FRET efficiency of the Closed state).

Understanding these physical meanings is crucial for interpreting the HMM output and relating it back to the molecular behavior of adenylate kinase.

In your HMM analysis, "the maximized log-likelihood of the model" (LL) refers to the highest probability of observing your entire set of experimental photon data, given a specific HMM model (defined by its parameters: Nstates, prior, transmat, obsmat) after that model's parameters have been optimally adjusted to fit the data.

Let's break it down:

Likelihood P(Data | Model):

For any given HMM (a specific number of states, and a specific set of initial state probabilities prior, transition probabilities transmat, and emission probabilities obsmat), you can calculate the probability of your observed sequence of photons (their arrival times and colors). This is the likelihood of the data given that specific model.

Your h2mm\_main.m function (specifically, the E-step and the forward algorithm within fwdback\_photonByphoton\_fast.m) calculates this likelihood for each iteration of the Expectation-Maximization (EM) algorithm.

Parameters of the Model:

For a chosen Nstates, the "parameters" of your HMM are the numerical values in the prior vector, the transmat matrix, and the obsmat matrix.

The EM algorithm is an iterative procedure designed to find the parameter values that maximize this likelihood P(Data | Model\_parameters).

Maximization (The EM Algorithm):

You start with an InitialGuess for prior, transmat, and obsmat.

E-step (Expectation): Given the current parameters, calculate the expected hidden state sequences and sufficient statistics (like expected number of transitions between states, expected time spent in each state before emitting a certain photon, etc.) that would explain your observed photon data. This step also yields the current log-likelihood of the data given the current parameters.

M-step (Maximization): Update the model parameters (prior, transmat, obsmat) to new values that increase the likelihood of observing those expected statistics calculated in the E-step.

You repeat the E and M steps. The log-likelihood LL is guaranteed to increase (or stay the same) with each EM iteration.

The algorithm stops when the increase in LL between iterations is very small (below your thresh) or when max\_iter is reached.

"Maximized" Log-Likelihood (The Best\_LL you report):

When the EM algorithm converges (or stops), the final LL value it reports is the maximized log-likelihood for that specific initial guess and that chosen number of states (Nstates). It represents the highest likelihood achieved by adjusting the model's parameters (prior, transmat, obsmat) through the EM process.

Since EM can get stuck in local optima, you run it multiple times with different InitialGuess values (your num\_initial\_guesses\_per\_N).

The Best\_LL for a given Nstates in your results\_summary table is the highest maximized log-likelihood found across all those different initial guesses for that particular Nstates.

In your case (from the table):

For Nstates = 2, after running the EM algorithm from 10 different starting points, the best set of parameters found for a 2-state model resulted in a maximized log-likelihood of approximately -3408. This means P(Your Photon Data | Best 2-State HMM) = exp(-3408).

For Nstates = 4, the best set of parameters found for a 4-state model resulted in a maximized log-likelihood of approximately -3376.2. This means P(Your Photon Data | Best 4-State HMM) = exp(-3376.2).

Since exp(-3376.2) is a larger probability than exp(-3408), the 4-state model provides a "better fit" to your data in terms of raw likelihood before any penalty for complexity is applied. AIC and BIC then use this Best\_LL to decide if the better fit of the 4-state model justifies its increased number of parameters.

So, when AIC/BIC formulas use LL, they are using this Best\_LL value – the peak of the likelihood function found by the EM algorithm for a model with a given number of states.