Algorithm 1 GLMHMM Class Pseudocode

```
procedure GLMHMM(N, n_states, n_features, n_outputs, max_iter, em_dist)
    self.N ← N
    self.n_states ← n_states
    self.n_features ← n_features + 1                ▷ Account for bias term
    self.n_outputs ← n_outputs
    self.max_iter ← max_iter
    if em_dist = gaussian then
        self.pdf ← multivariate_normal.pdf
        self.w ← Random initialization of weights with bias
        self.covariances ← Small identity matrices for each state
    else
        self.pdf ← None
    end if
    self.transition_matrix ← Uniform distribution for states
end procedure
procedure DIST_PARAM(w_k, X)
    Compute pre_act ← X · w_k
    Return tanh(pre_act)
end procedure
procedure NEGLOGLI(w_k, X, Y, γ_k)
    w_k ← Reshape weights if needed
    Compute θ_k ← dist_param(w_k, X)
    Compute likelihood list: ll_list ← γ_k · log(dist_pdf(Y, θ_k))
    Return −∑(ll_list)
end procedure
procedure DIST_PDF(Y, θ_k, other_params)
    if PDF is Gaussian then
        Compute likelihood using multivariate normal distribution
    else
        Raise Exception: "No PDF defined"
    end if
    Return likelihood
end procedure
procedure COMPUTE_LIKELIHOOD(X, Y)
    Initialize likelihood list ll
    for each state k in n_states do
        w_k ← self.w[k]
        θ_k ← dist_param(w_k, X)
        ll[k] ← dist_pdf(Y, θ_k, covariances[k])
    end for
    Return likelihood array ll
end procedure
procedure PREDICT(X, Y)
    Augment X with bias term
    Initialize log probabilities and previous states
    for each time t in X do
        for each state j do
            Compute log_prob[t, j] ← max(log probabilities of previous states)
            Track prev_states[t, j]
        end for
    end for
    Perform backtracking to retrieve the most likely sequence of states
    Return state sequence
end procedure
```

procedure GENERATE_DATA($n\_samples$)
Initialize $X, Y,$ states
Sample initial state and observation
for each time step t **do**
Sample next state based on transition probabilities
Sample observation $Y[t]$ using current state's emission distribution
end for
Return $X, Y,$ states
end procedure
procedure UPDATE_TRANSITIONS($y, \alpha, \beta, cs, A, \phi$)
Initialize $\xi \leftarrow \mathbf{0}$ with shape $(N-1, K, K)$
for $i \leftarrow 0$ to $N-2$ **do**
Compute beta_phi $\leftarrow \beta[i+1, :] \odot \phi[i, :]$
Reshape $\alpha[i, :]$ into alpha_reshaped $\in R^{K\times 1}$
Update $\xi[i, :, :] \leftarrow \frac{(\text{beta\_phi} \times \text{alpha\_reshaped}) \odot A}{cs[i+1]}$
end for
Compute $\xi_n \leftarrow \sum_{\text{over } i} \xi[i, :, :]$
Compute $\xi_{kn} \leftarrow \sum_{\text{over rows}} \xi_n$ and reshape to $R^{K\times 1}$
Compute $A_{\text{new}} \leftarrow \frac{\xi_n}{\xi_{kn}}$
Return A_{new}
end procedure
procedure UPDATE_OBSERVATIONS(y, x, w, γ)
if y is 1-dimensional **then**
Reshape y to $R^{N\times 1}$
end if
Initialize $\phi \leftarrow \mathbf{0}$ with shape (N, K)
for $z_k \leftarrow 0$ to $K-1$ **do**
Print z_k
Compute $w[z_k], \phi[:, z_k] \leftarrow$ GLM_FIT($x, w[z_k], y,$ covariances$[z_k]$, gammak $= \gamma[:, z_k]$)
Compute $\theta_k \leftarrow$ DIST_PARAM($w[z_k], x$)
Compute residuals residuals $\leftarrow y - \theta_k$
Update covariances$[z_k] \leftarrow$ Covariance of residuals
end for
Return w, ϕ
end procedure
procedure GLM_FIT($x, w_k, y,$ otherparamk, compHess, gammak, gaussianPrior)
Flatten w_k to w_{flat}
Define opt_log $\leftarrow$ NEG_LOG_LIKELIHOOD($w, x, y,$ gammak, otherparamk)
Optimize w_{flat} using L-BFGS-B and store result in w_{opt}
Reshape w_{opt} to $D \times$ output_dim and append a zero row
Compute $\theta_k \leftarrow$ DIST_PARAM(w_k, x)
Compute $\phi \leftarrow$ DIST_PDF($y, \theta_k,$ otherparamk)
Return w_k, ϕ
end procedure
procedure UPDATE_INIT_STATES(γ)
Compute $\pi \leftarrow \frac{\gamma[0,:]}{\sum \gamma[0,:]}$
Return π
end procedure

procedure UPDATE_PARAMS($y, x, \gamma, \beta, \alpha, cs, A, \phi, w, \text{fit\_init\_states}$)
    Update $A \leftarrow$ UPDATE_TRANSITIONS($y, \alpha, \beta, cs, A, \phi$)
    Update $w, \phi \leftarrow$ UPDATE_OBSERVATIONS(y, x, w, γ)
    if fit_init_states is True **then**
        Update $\pi_0 \leftarrow$ UPDATE_INIT_STATES(γ)
    end if
    Return A, w, ϕ, π_0
end procedure
procedure FIT($y, x, A, w, \pi_0, \text{fit\_init\_states}, \text{maxiter}, \text{tol}, \text{sess}, B$)
    Augment x with a column of ones for bias term
    Initialize lls with NaNs for storing log-likelihood values
    if no session boundaries provided **then**
        Set $sess \leftarrow [0, N]$
    end if
    Compute initial ϕ using w and emission probability functions
    for $n \leftarrow 0$ to maxiter-1 **do**
        Initialize α, β, cs, posterior state probabilities, most probable states
        $ll \leftarrow 0$
        for each session in sess **do**
            Compute $ll_s, \alpha_s, cs_s \leftarrow$ FORWARD_PASS(y[session range]$, A, \phi, \pi_0$)
            Compute β_s, posterior states, most probable states $\leftarrow$ BACKWARD_PASS(y[session range]$, A, \phi, \alpha_s, cs_s$)
            Accumulate log-likelihood $ll \leftarrow ll + ll_s$
        end for
        Store ll in $lls[n]$
        Update $A, w, \phi, \pi_0 \leftarrow$ UPDATE_PARAMS(y, x, posterior states$, \beta, \alpha, cs, A, \phi, w$, fit_init_states)
        if convergence criteria met based on tolerance **then**
            Break
        end if
    end for
    Return lls, A, w, π_0
end procedure
procedure FORWARD_PASS(y, A, ϕ, π_0)
    Initialize $\alpha, \alpha_{\text{prior}}, cs$
    if no initial probabilities provided **then**
        Set $\pi_0 \leftarrow$ uniform distribution over states
    end if
    Compute initial α, cs using $\phi[0, :]$ and π_0
    for $t \leftarrow 1$ to $N - 1$ **do**
        Compute $\alpha_{\text{prior}}[t]$ from $\alpha[t-1]$ and A
        Compute $cs[t]$ and normalize $\alpha[t]$
    end for
    Compute log-likelihood $ll \leftarrow \sum \log(cs)$
    Return $ll, \alpha, \alpha_{\text{prior}}, cs$
end procedure
procedure BACKWARD_PASS(y, A, ϕ, α, cs)
    Initialize β with 1 for the last time step
    for $t \leftarrow N - 2$ to 0 (backwards) **do**
        Compute $\beta[t]$ using $\phi[t+1, :]$, A, and $cs[t+1]$
    end for
    Compute posterior state probabilities posterior $\leftarrow \alpha \odot \beta$
    Decode most probable states states $\leftarrow \text{argmax}(\text{posterior}, \text{axis} = 1)$
    Return posterior$, \beta,$ states
end procedure