Key Derivations and Expansions

1. Causal Gaussian Process (GP) Kernel

The paper discusses the use of a dynamic causal GP model but does not provide explicit details about the kernel. Based on the code (e.g., `CausalRBF` in `causal_kernels.py`), the kernel seems to extend the standard RBF kernel to capture temporal dependencies. Here is the derivation:

Standard RBF Kernel:

$$k_{\mathrm{RBF}}(x, x') = \sigma^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right),$$

where:

- σ^2: Kernel variance.
- ℓ: Length scale parameter.

Causal Extension: The `CausalRBF` kernel introduces dependencies on time and causal structure:

$$k_{\mathrm{CausalRBF}}(x, x') = k_{\mathrm{RBF}}(x, x') \cdot \phi(t, t'),$$

where:

- $\phi(t, t')$ encodes temporal causality between time points t and t'. This may involve a decay term to capture the diminishing influence of older time steps.

Implementation Insight: The kernel is computed pairwise for all time indices, and its parameters are optimized during GP training. This causal extension is not fully described in the paper but is crucial for implementing DCBO.

2. Transfer of Interventional Information

The paper mentions the transfer of interventional data across time steps but does not elaborate on the mechanics. From the code (`_update_sufficient_statistics` in `root.py`), this involves updating the mean and variance functions for each temporal step.

Derivation: Given:

- $X_t^{\mathrm{obs}}, Y_t^{\mathrm{obs}}$: Observational data at time t.

- $X_t^{\text{int}}, Y_t^{\text{int}}$: Interventional data at time t.

The GP posterior mean and variance for an unobserved point x_* are:

$$\mu_t(x_*) = K_{*t}(K_{tt} + \sigma_n^2 I)^{-1} Y_t,$$
$$\Sigma_t(x_*) = K_{**} - K_{*t}(K_{tt} + \sigma_n^2 I)^{-1} K_{t*},$$

where:

- K_{tt}: Kernel matrix for training points.
- K_{*t}: Kernel vector between x_* and training points.

The interventional update integrates $X_t^{\text{int}}, Y_t^{\text{int}}$ to refine $\mu_t(x_*)$ and $\Sigma_t(x_*)$. This dynamic update is central to transferring knowledge across time steps.

3. Causal Expected Improvement (CEI)

The paper introduces the CEI acquisition function but provides limited details on its derivation. Here is a step-by-step derivation:

Expected Improvement (EI): For a target variable Y, the standard EI is:

$$\text{EI}(x) = \mathbb{E}\left[\max(0, y^* - f(x))\right],$$

where y^* is the best observed value and $f(x)$ is the GP mean prediction.

Causal Extension: In CEI, the improvement accounts for causal influences:

$$\text{CEI}(x) = \mathbb{E}\left[\max(0, y^* - f(x)) \mid \text{Causal Graph}\right].$$

This requires integrating over the causal relationships encoded in the SEM:

$$\text{CEI}(x) = \int_{\mathcal{X}} \text{EI}(x) \cdot p(x \mid \text{SEM})\, dx.$$

Implementation: The code evaluates CEI numerically using `evaluate_acquisition_functi` which dynamically updates the SEM and acquisition values based on observational and interventional data.

4. SEM Parameter Estimation

The SEM estimation process (`fit_arcs` in `sem_estimate.py`) is not described in the paper. It fits GP models to each arc in the causal graph.

Derivation: For a given node X_i with parents P_i:

$$X_i = f(P_i) + \epsilon_i,$$

where:

- $f(P_i)$: A function learned using GPs.
- ϵ_i: Gaussian noise.

The likelihood of the data is:

$$p(X_i \mid P_i) = \mathcal{N}(f(P_i), \sigma^2).$$

Maximizing this likelihood estimates the GP hyperparameters (ℓ, σ^2) for each arc. The estimation integrates over all nodes to construct the SEM.

5. Dynamic Updating of SEM

The dynamic aspect involves continuously updating the SEM with interventional data at each time step.

Process:

1. Use observational data to initialize the SEM.
2. Update the GP models for each arc using interventional data:

$$f(P_i) \leftarrow \text{GP update with } (X_i^{\text{int}}, P_i^{\text{int}}).$$

3. Refine the causal graph structure based on time-evolving dependencies.