What is computational interaction design?

Computational interaction would typically involve at least one of:

I. an **explicit mathematical model** of user-system behavior;
II. a way of updating that model with **observed data** from users;
III. an **algorithmic element** that, using this model, can directly synthesise or adapt the design;
IV. a way of **automating and instrumenting** the modeling and design process;
V. the ability to **simulate or synthesise** elements of the expected user-system behavior.

Computational interaction often involves elements from machine learning, signal processing, information theory, optimisation, inference, control theory and formal modelling.
Contrast with traditional approaches

**Traditional HCI**
more design ingenuity
better elicitation and design techniques
stronger evaluation

*Example:*
A designer invents a mid-air interaction, logs performance with users, and performs a statistical analysis. The designer improves the design informed by the evaluation results.

*No design work was automated.*
*No explicit model.*
*Data influenced design only through designer.*

**Computational HCI**
improved modeling
better data collection
more powerful algorithms
increased computational power

*Example:*
A designer builds a model of pointing behaviour in mid-air from data. An algorithm is used to optimise the spacing of targets.

*Design work performed by algorithm.*
*Explicit modeling.*
*Data directly influenced design.*
Why do computational interaction design? (I)

Automation, data and models can supplant hand-tweaking

=> reduce design time of interfaces.

Better models can better predict how interactions evolve

=> build more robust and efficient interfaces.

Structure can be learned rather than dictated

=> better tailored interfaces: to users, contexts, devices.

Fundamental processes that generalise to new contexts

=> harness new technologies quickly.
Why do computational interaction design? (II)

Strong models can predict much of expected user behavior
   =>  reduce the evaluation burden.

HCI problems can be defined formally
   =>  increases our ability to reason rather than blind experimentation

Algorithmic design can support designers in tedious tasks
   =>  focus on creative aspects of design
Themes

- **Optimisation**: Delegating design decisions to automatic optimization, rather than fine tuning every detail by hand.
  - This requires defining objective functions to measure performance, setting models of interfaces and selecting and running optimisation algorithms
  - This draws on optimisation theory and models of human performance.

- **Inference**: Recovering intention from measured signals at input devices.
  - This involves creating mappings between measurements and actions that go beyond simple hard-coded rules, by learning from observed data and inferring intention probabilistically
  - This draws on machine learning and probabilistic inference.

Underpinned by

- **Modeling**: Executable approximations of system, user and system-user behaviour