A Population Approach to Ubicomp System Design

Matthew C. Higgs*, Mark Girolami*, Matthew Chalmers, Muffy Calder, Alistair Morrison, Oana Andrei, Marek Bell, Scott Sherwood, & John Rooksby

*Center for Computational Statistics and Machine Learning
Department of Statistical Science
University College London
Contents

Introduction
  Project Description
    Vision and Ambition

Statistical Analysis
  Analysis of User Traces
    User life-time length

Future Work
  Future Apps
    Socio-technical
    Predicted Legacy
Project

Communication.
Existing Infrastructure

Contributions.

- Formal Modelling
- Scottish Premier League
- Living PlanIT
- Edinburgh Festival
- App Development
- Data Logging
- Industrial Collaboration
- Statistical Modelling
- Existing Infrastructure

Contributions.
Existing Infrastructure

Contributions.

Formal Modelling

Statistical Modelling

App Development

Post Factory

Match FFIT

App Tracker

Data Logging

Industrial Collaboration

Hungry Yoshi
Existing Infrastructure

Contributions.
Existing Infrastructure

Work flow.

- Formal Modelling
- App Development
- Statistical Modelling
- Data Logging
- Industrial Collaboration
Contents

Introduction
  Project Description
  Vision and Ambition

Statistical Analysis
  Analysis of User Traces
  User life-time length

Future Work
  Future Apps
  Socio-technical
  Predicted Legacy
Adaptive Design

The ideal.

- Observe software in the wild.
- Initiate a positive change based on observation.

The ideal is ambiguous about what is observed, what change is made, and who makes the change.
The emergence of the “app store” market has enabled researchers to run worldwide ubicomp trials with huge numbers of users.
Software in the Wild

- Study type.

- Post Factory
- Trial methodology
- Hungry Yoshi

- Modularity
- Ubiquity

- App Development
- Match FFIT
- App Tracker
Software in the Wild

- Study type.

Modularity

Trial methodology

Ubiquity

Data Logging

SGLog

Data in the Wild

Software in the Wild Study type.
Data Analysis
Vision Summary

Trials:
• Create socio-technical environments and persuade people to immerse themselves.

Statistical:
• Infer the structure and dynamics in the use and evolution data of software populations.

Ambition:
• Develop visualisation tools and formal methods to guide “developers” in their design decisions.
Analysis of User Traces

Consider a string of symbols

\[ a = a_1 a_2 \cdots a_N. \]  

(1)

Where each symbol \( a_i \) takes values in a finite set \( A \).

Consider a set of strings

\[ A = \{ a_u, u = 1, \ldots, M, |a_u| = N(u) \in \mathbb{N} \}. \]  

(2)

Assume \( A \) to be a set of user traces, where \( A \) represents a set of possible actions.
Analysis of User Traces

Given a set of user traces $A$, we want to:

- Characterise the *natural behaviours* of the population.
- Summarise the population based on these behaviours.

“Natural behaviours” are frequently occurring patterns of actions.

A “summary” of the population is a representation of the population in a low dimensional space spanned by latent behavioural semantics.

We use *Probabilistic Latent Semantic Analysis*. 
Analysis of User Traces

PLSA consists of:

- A directed graph (digraph) \( G(\mathcal{A}, E) \) representing the set of possible action transitions in the app.
- A set of \( K \) transition matrices \( \{T_k, k = 1, \ldots, K\} \) over the (actionable) nodes of the graph.
- A mixture weighting \( \theta = (\theta_1, \ldots, \theta_K) \) for each user.

We assume each string \( a \) is generated by:

- Moving from \( a_i \) to \( a_{i+1} \) using transition \( T_k \) with probability \( \theta_k \).

An *admixture* of (first-order) discrete-time Markov chains.
Simplified Hungry-Yoshi

(a) Main-menu screen-shot.
(b) Yoshi screen-shot.
(c) Plantation screen-shot.

Symbol
Yoshi Feed Plant Pick
Description
View a yoshi.
Feed a yoshi.
View a plant.
Pick a fruit.

(d) Symbol table and descriptions of corresponding in-app events.

Yoshi Plant Plant Yoshi Plant Pick Pick Pick Yoshi Feed Feed Feed Plant Pick Pick

(e) An example of a typical user-trace.
Hungry-Yoshi Digraph

Figure: Yoshi, Feed, Plant, Pick enumerated as \{1, 2, 3, 4\}.
Hungry-Yoshi PLSA ($K = 2$)

(a) Digraph 1.

(b) Digraph 2.

(c) Population weights.
Successful vs. Unsuccessful Strategies

What is the criteria for success?
Successful vs. Unsuccessful

- User is successful if they complete a number of tasks.
- Developer is successful if a number of users complete a number of tasks.

Figure: Population weights with trace-lengths.
PLSA Summary

- Maximum Likelihood PLSA is prone to over-fitting, but do we care.
- How will developers use the results. (Personalisation vs fragmentation).
- How will formal methods use these results.
- How do we choose $K$.
- Extend to MDPs, identify “states” and latent reward mechanisms.
Contents

Introduction
  Project Description
  Vision and Ambition

Statistical Analysis
  Analysis of User Traces
  User life-time length

Future Work
  Future Apps
  Socio-technical
  Predicted Legacy
User life-time length

Log–log PF population data plot.

Figure: Plot of Post-Factory population data on a log-log scale. $N(t)$ denotes number of users who performed at least $t$ actions.
Coin-toss game model

- **Constant bias** (CB). Every player in every round uses the same coin with bias $p \in [0, 1]$.
- **Empirical bias** (EB). Every player in each round uses the same coin, but in each round a new coin with bias $p_t \in [0, 1]$ is given to all players.
- **Functional bias** (FB). The bias $p_t$ is assumed to have a functional form

$$p_t = (1 - p_0)(1 + e^{-t/\alpha}) + p_0,$$

where $p_0 \in [0, 1]$ is an initial probability, and $\alpha > 0$ a scale parameter.
Figure: Bias parameters for each CTG model, estimated using the full Post-Factory population data set.
Coin-toss game model

Figure: Mean path plots for each of the CTG models.

<table>
<thead>
<tr>
<th>K-fold</th>
<th>CB-model</th>
<th>EB-model</th>
<th>FB-model</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-fold</td>
<td>$71.9 \pm 17.4$</td>
<td>$17.5 \pm 12.2$</td>
<td>$14.8 \pm 11.6$</td>
</tr>
<tr>
<td>5-fold</td>
<td>$13.1 \pm 1.74$</td>
<td>$4.44 \pm 1.57$</td>
<td>$4.25 \pm 1.52$</td>
</tr>
<tr>
<td>10-fold</td>
<td>$4.12 \pm 0.48$</td>
<td>$1.97 \pm 0.46$</td>
<td>$1.93 \pm 0.45$</td>
</tr>
</tbody>
</table>

Figure: Estimates of MSE using $K$-fold CV. Mean MSE $\pm$ one std, from 10,000 reps.
CTG Summary

• The best performer is the FB-model with bias

\[ p_t = (1 - p_0)(1 + e^{-t/\alpha}) + p_0. \]  \hspace{1cm} (4)

• This is a *psychometrically* motivated *choice model*.
• \( p_0 \) can be thought of as a function of the properties of the app at start up.
• \( \alpha \) can be thought of as a *reward rate* that determines how “attached” the user becomes with time.
CTG Future

- The model can be used in A/B testing of user experiences.
- The model can be extended to the regression setting where $-t/\alpha$ is replaced with a function of user specific context.
- All latent decisions of the user can be modelled using similar *choice analysis* methods with psychometric interpretations.
Post Factory

Modularity study.

Figure : Post Factory Screen Shots.

Exploration-exploitation model.
App Tracker

Ubiquity study.

Figure : App tracker analytics.

PLSA.
Match FFIT

Audience permeation.

Figure: Match FFIT logo.

Slowly introduce modularity.
Contents

Introduction
  Project Description
  Vision and Ambition

Statistical Analysis
  Analysis of User Traces
  User life-time length

Future Work
  Future Apps
  Socio-technical
  Predicted Legacy
Figure: “Friends phones now” sketch.
Stereotypes

Experiments that create specific socio-technical environments and ask humans to enter them.

(Milner’s ubicomp “vision” 2006)

- How do we attach labels to the latent behavioural semantics.
- How will people behave if these labels are made public.
Contents

Introduction
  Project Description
  Vision and Ambition

Statistical Analysis
  Analysis of User Traces
  User life-time length

Future Work
  Future Apps
  Socio-technical
  Predicted Legacy
SGTracker

Adaptive logging infrastructure.
SGViz-2.0
Sophisticated visualisation tools.
Population Models

Treatment of *class* as stochastic.