Open access • Posted Content • DOI:10.1101/2021.08.08.455583

The Normative Modeling Framework for Computational Psychiatry — Source link

Saige Rutherford, Saige Rutherford, Seyed Mostafa Kia, Seyed Mostafa Kia ...+13 more authors

Institutions: University of Michigan, Radboud University Nijmegen, Utrecht University, University of Oslo ...+4 more institutions

Published on: 10 Aug 2021 - bioRxiv (Cold Spring Harbor Laboratory)

Topics: Normative model of decision-making, Normative and Population

The Normative Modeling Framework for Computational Psychiatry

Saige Rutherford[1,2,3], Seyed Mostafa Kia[1,2,4], Thomas Wolfers[5,6], Charlotte Fraza[1,2], Mariam Zabihi[1,2], Richard Dinga[1,2], Pierre Berthet[5,6], Amanda Worker[7], Serena Verdi[8,9], Henricus G. Ruhe[10*], Christian F. Beckmann[1,2,11*], Andre F. Marquand[1,2,7*]

[1] Donders Institute for Brain, Cognition, and Behavior, Radboud University, Nijmegen, the Netherlands
[2] Department of Cognitive Neuroscience, Radboud University Medical Center, Nijmegen, the Netherlands
[3] Department of Psychiatry, University of Michigan, Ann Arbor, MI, United States
4 Department of Psychiatry, Utrecht University Medical Center, Utrecht, the Netherlands
[5] Department of Psychology, University of Oslo, Oslo, Norway
[6] Norwegian Center for Mental Disorders Research, University of Oslo, Oslo, Norway
[7]Department of Psychological Medicine, Institute of Psychiatry, Psychology and Neuroscience, King's College London, London, United Kingdom
[8]Centre for Medical Image Computing, Medical Physics and Biomedical Engineering, University College London, London, UK
[9]Dementia Research Centre, UCL Queen Square Institute of Neurology, London, United Kingdom
[10] Department of Psychiatry, Radboud University Medical Center, Nijmegen, the Netherlands
[11] Centre for Functional MRI of the Brain, University of Oxford, Oxford, United Kingdom
[*]Contributed equally to senior author

Abstract

Normative modeling is an emerging and innovative framework for mapping individual differences at the level of a single subject or observation in relation to a reference model. It involves charting centiles of variation across a population in terms of mappings between biology and behavior which can then be used to make statistical inferences at the level of the individual. The fields of computational psychiatry and clinical neuroscience have been slow to transition away from patient versus "healthy" control analytic approaches, likely due to a lack of tools designed to properly model biological heterogeneity of mental disorders. Normative modeling provides a solution to address this issue and moves analysis away from case-control comparisons that rely on potentially noisy clinical labels. In this article, we define a standardized protocol to guide users through, from start to finish, normative modeling analysis using the Predictive Clinical Neuroscience toolkit (PCNtoolkit). We describe the input data selection process, provide intuition behind the various modeling choices, and conclude by demonstrating several examples of down-stream analyses the normative model results may facilitate, such as stratification of high-risk individuals, subtyping, and behavioral predictive modeling.

Keywords: normative modeling, computational psychiatry, individual differences, precision medicine, software tutorial, lifespan neuroscience, brain growth charting

Introduction

Clinical neuroscientists have recently acknowledged two realities that have disrupted the way research is conducted: first, that to understand individual differences it is necessary to move away from group average statistics [1–7] and, second, that the classical diagnostic labels of psychiatric disorders are not clearly represented in the underlying biology[8–11]. Despite this awareness and an increasing interest in quantifying individual differences, the field has been slow to transition away from case-control comparisons that aim to contrast patient versus healthy control groups and assume that clinical groups are distinct and homogenous. A key barrier that has impeded progress is a lack of alternative analysis methods, designed to model variation across individuals, also known as heterogeneity[12]. Nearly all existing techniques for connecting the brain to behavior operate at the group-level and provide no path to individual-level inference[13–15]. Normative modeling is a framework for understanding differences at the level of a single subject or observation while mapping these differences in relation to a reference model (Figure 1). It involves charting centiles of variation across a population in terms of mappings between biology and behavior, which can then be used to make statistical inferences at the level of the individual, akin to the use of growth charts in pediatric medicine (Figure 1A). The practice of normative modeling in clinical neuroscience was developed to provide additional information beyond what can be learned from case-control modeling approaches. Case-control thinking assumes that the mean is representative of the population, when it may not be (e.g., if the clinical population is diffuse or comprised of multiple sub-populations). Therefore, normative modeling has become a leading tool for precision medicine research programs and has been used in many clinical contexts[16].

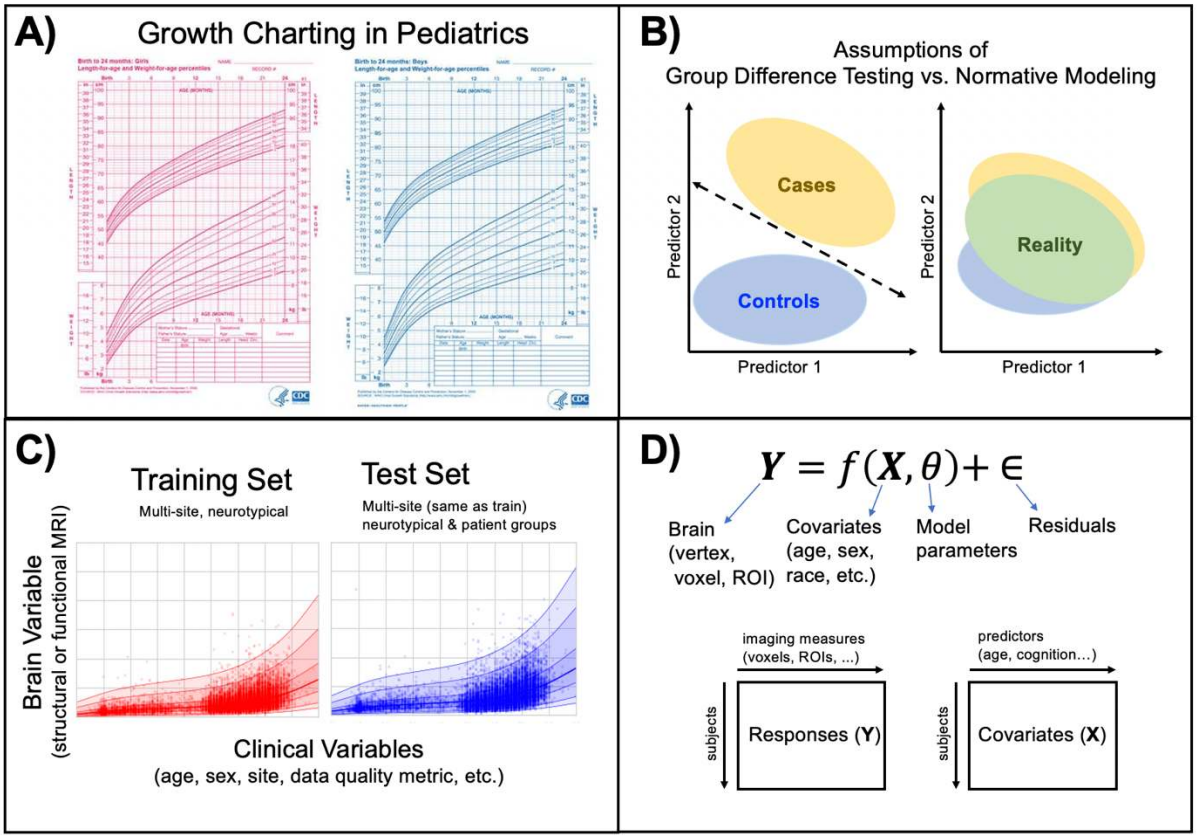

Figure 1 Conceptual Overview of Normative Modeling. A) Classical example of normative modeling: the use of height and weight growth charting in pediatrics. **B)** Case-control models (left) theoretically make assumptions that there is a boundary that can separate groups and that there is within-group homogeneity. In reality (right), there is nested variation across controls and patient groups and within-group heterogeneity, resulting in unclear separation boundaries. Normative modeling is well equipped to handle this reality. **C)** An example application of normative modeling in computational psychiatry using neuroimaging data. Mean cortical thickness (y-axis) is predicted from age (x-axis) using a training set consisting of multi-site structural MRI from neurotypical controls and a test set consisting of neurotypical controls and patient groups. Every dot indicates the deviation score for a single individual from normal development. **D)** Regression model equation and design matrix setup for the model shown in panel C.

Neuroscience has historically brought together scientists from diverse educations, for example, some from a clinical background and others having a mathematics background. The interdisciplinary nature introduces a challenge in bridging the gap between technical and clinical perspectives. This is a key challenge that aligns with the aims of the open-science movement and brain-hack community[17], in other words, to distill the essential components of the analytic workflow into a consistent and widely applicable protocol. This helps to avoid 'research debt', i.e., a lack of ideas being digested[18]. This distiller mindset is crucial for confronting research debt

and embracing paradigm shifts in thinking, such as moving from case-control comparisons to the normative modeling framework.

The purpose of this work is to distill the methods of normative modeling, an advanced analysis technique, into an actionable protocol that addresses these challenges in that it is accessible to researchers within the diverse field of clinical neuroscience. We distill the essential components of a normative modeling analysis and provide a demonstrative analysis from start to finish using the Predictive Clinical Neuroscience Toolkit software. We describe the input data selection process, give an overview of the various modeling choices, and conclude by demonstrating several examples of downstream analyses the normative model results may facilitate, such as stratification of high-risk individuals, subtyping, and behavioral predictive modeling.

Development of the protocol

Normative modeling has a long history that relates to statistics and measurement theory and has many applications from medicine to economics to neuroscience. Familiar use cases of normative modeling include growth charting in pediatrics, neurocognitive tests, and interpreting graduate school test score percentiles (i.e., scoring 90th percentile on the MCAT). The mathematical and computational development of normative modeling has been fine-tuned[19–22] and currently exists as an open-source software python package, the Predictive Clinical Neuroscience toolkit (PCNtoolkit), which we focus on in this manuscript. This toolkit implements many commonly used algorithms for normative modelling and supports multiple industry standard data formats (e.g., NIFTI, CIFTI, text formats). Extensive documentation has been written to accompany this protocol and is available online through read the docs. This includes tutorials with sample data for all algorithm implementations, a glossary to help new users understand the jargon associated with the software, and a frequently asked questions page. An online forum for communicating questions, bugs, feature requests, *etc.* to the core team of PCNtoolkit developers is also available. We have developed these open-source resources to promote and encourage individual differences research in computational psychiatry using normative modeling.

Applications and comparison with other methods

Normative modeling has been applied to many research questions in computational psychiatry and other fields, including in autism spectrum disorder[23–25], attention deficit

hyperactive disorder[26,27], Alzheimer's disease[28], bipolar disorder, and schizophrenia[29–31]. Crucially, these applications have shown that normative modelling can detect individual differences both in the presence of strong case-control differences[30] and in their absence[24]. This highlights the value and complementary nature of understanding individual variation relative to group means. These applications have primarily focused on predicting regional structural or functional neuroimaging data (i.e., biological response variables) from phenotypic variables (i.e., clinically relevant covariates) such as age and sex. Age creates a natural, time-varying dimension for mapping normative trajectories and is well suited to applications in which deviations of an individual manifest from a typical trajectory of brain development or ageing. However, other phenotypes that have been used in neuroimaging predictive modeling studies such as general cognitive ability[32,33], social cognition, or sustained attention[34,35] are also attractive possibilities to use as covariates, thereby defining axes for observing deviation patterns. Normative modeling has also been used to learn mappings between reward sensitivity and reward related brain activity[36].

It is important to emphasize that normative modeling is a general regression framework for mapping sources of heterogeneity, refocusing attention on individual predictions rather than group means (e.g., diagnostic labels), and detecting individuals who deviate from the norm. Therefore, it is not limited to a specific algorithm or mathematical model, although we recommend certain algorithms based on the research question and available input data. The algorithms in the PCNtoolkit tend to favor Bayesian over frequentist statistics, as there are certain features of Bayesian approaches that facilitate better normative modeling estimation. For example, having a posterior distribution over the parameters help to better separate different sources of uncertainty, e.g., separating variation ('aleatoric uncertainty') from modelling (or 'epistemic') uncertainty. These different use cases of normative modeling (algorithm selection, predicting brain from behavior or behavior from the brain) are explained in-depth in the experimental design section.

There is a long history of using regression methods to learn mappings between brain and behavior[37,38]. Brain Basis Set modeling (BBS)[14,39,40], Connectome predictive modeling (CPM)[13,41], and canonical correlation analysis (CCA)[42,43] have become mainstream methods for linking brain and behavior. These methods have demonstrated the feasibility of brain-behavior mapping and laid the foundation for individual differences research to thrive. While these

approaches have generated much curiosity and excitement, they are limited in their ability to provide inference at the level of the individual and only provide estimates of the mean (i.e., without associated centiles of variation). Most papers using these tools only report the overall predictive model performance, collapsing information across hundreds or thousands of people into a single number (e.g. model accuracy or regression performance)[40,41,44,45]. The normative modeling framework takes these ideas a step further to quantify and describe how individuals differ. Case-control inference (e.g., mass univariate group t-testing) and classification (patient *vs.* control) examples are perhaps the most interesting comparison to the normative modeling framework. Normative models reveal a different side of the data -- that the classical diagnostic labels of psychiatric disorders are not clearly represented in the underlying biology, meaning patient groups are not well defined by a unifying neurosignature -- and provide clear evidence for the limitations of case-control paradigms. Brain age models are also in the same family as normative models but generally have a narrower focus on interpreting accelerated/decelerated aging[46,47] or improving prediction accuracy[48]. Brain age models only allow for interpreting centiles of variation in terms of age, which is limited and does not have a clear interpretation in terms of biological variation across individuals.

Overview of the procedure

Experimental design

There are many choices and considerations that should be carefully planned before embarking on a normative modeling analysis – the decision points can be grouped into the following themes: data selection, data preparation, algorithm/modeling, and evaluation/interpretation (Figure 2). Creating the training dataset that will serve as the "normative" reference curve is the first important decision. Ideally, the training dataset will be a large and representative sample, and the included subjects should not be missing vital demographic (age, sex) or biological (neuroimaging) data. However, data imputation may be used if necessary but should be used cautiously. In most research studies, data are missing not at random, and we interpret more than just mean effects. In this case, mean imputation will bias results and other forms of imputation should be considered[49,50]. It is important that the reference cohort provides good coverage (complementary covariates) of the test set (e.g., clinical) population. For example, it would not be sensible to model age ranges of childhood and adolescence in the reference cohort and have the test cohort consist of late adulthood ages.

It is rare for a single scanning site to acquire large enough samples that are an accurate representation of the general population. Therefore, it is typically necessary to pool data obtained across multiple MRI centers. Some projects, such as the ABCD study[51], have begun to harmonize scanning protocols because multi-site pooling was planned prior to data collection. In contrast, other projects, such as ENIGMA[52], combine data post-collection and not have harmonized scanning sessions prior to data collection. If possible, to eliminate additional sources of variance, multi-site pooled data should be preprocessed using identical pipelines and software versions. However, due to data sharing restrictions and privacy concerns regarding health data, raw data may be unavailable, making pre or post data collection harmonization efforts impossible. Data harmonization techniques, such as COMBAT[53–56], aim to remove site-related variance from the data as a preprocessing step before further analyses are run. There are some issues with harmonization, principally that all sources of variance that are correlated with the batch-effects (i.e., site-related variance) are removed which can unintentionally remove important, unknown, clinically relevant variance from the data. COMBAT also requires that the user have access to all the data when harmonizing which does not bode well with data privacy concerns. We therefore do not recommend users focus on data harmonization techniques when preparing their data sets for normative modeling. Hierarchical Bayesian Regression (HBR)[21,22] implemented in the PCNtoolkit has been thoroughly developed and tested to address these challenges when using multi-site data in normative modeling. HBR estimates site-specific mean effects and variations in the normative model estimation stage using a Bayesian hierarchical model, which produces site-agnostic deviation scores (z-statistics). This distinction between harmonization techniques (i.e., COMBAT) and HBR-normative modeling is very important when using deviation scores as features in subsequent interpretation analyses, as harmonization has been shown to overexaggerate confidence in downstream analyses[57].

The next choice should be regarding which covariates to include. One of the main criteria to include a covariate is the relevance to the posed research question. In normative modeling, usually we are interested in studying the deviations from the norm of the population, in other words, we are more interested in residuals. Thus, when we include a covariate in the design matrix for estimating the normative model, we are mainly interested in removing its effect from the residuals (thus deviations) than investigating its effect on the neuroimaging variable. Normative modeling is a tool to study unknowns (that are encoded in the deviations). To do so,

we need to first account for known variation in the data by regressing them out of the data (thus we include the knowns in the covariates), and then we interpret the residual variation in the deviation scores. For example, if you want to know the effect of smoking on the ventral striatum, that is not confounded by other substance use, you should include substance use variables (e.g., drinks per week, etc.) in the covariate matrix, estimate the normative model, and then correlate the ventral striatum deviation score (that has the effect of drinking removed from it) with smoking frequency. When pooling data from multiple sites, the available measures across sites may influence the selection of covariates because ideally, the variables should be consistent across sites. For example, you should not use different versions of a cognitive test, as they could test for different dimensions of general cognitive ability. For neurodevelopmental or lifespan model, the suggested minimum covariates to include are age, sex, site (using random- or fixed-effects), and optionally a metric of data quality (i.e., mean framewise displacement or Freesurfer Euler number). Modeling site is very important; however, an exhaustive explanation is outside the scope of this protocol but see [20,21] for an in-depth account of modeling site variation. Diagnostic labels could also be included as covariates to utilize the variance explained by these labels without constraining the mapping to only reflect case-control differences. Furthermore, additional biological covariates could also be included, for example blood biomarkers, or structural brain measures if predicting functional brain measures. Additional or alternative covariates may include other demographics (race, ethnicity, gender, education level, marital status, household income) and cognitive variables.

Next, it is necessary to decide on the resolution of the input brain data. The resolution of predictions is important to consider while keeping in mind the increasing computational complexity with modeling smaller units. Vertex or voxel-level modeling of brain data provides high-resolution deviation maps. Still, ROI-level modeling may allow for easier interpretation/visualization of the output deviation maps and will have a lower penalty in multiple comparison correction (if doing post-hoc analysis) on the deviation maps.

After the data have been carefully chosen and curated, it is time to move onto the normative modeling implementation. There are several algorithms for implementing a normative model including Gaussian process regression[58], Bayesian linear regression[19,59], hierarchical Bayesian regression[21,22], generalized additive models of location, scale, and shape[20], neural processes[60], random feature approximation[61], quantile regression[62] and many of these are

implemented in the PCNtoolkit software package. The algorithms have different properties depending on their ability to model non-linear effects, scaling to large data sets (in terms of computation time), handling of random or fixed effects (e.g., to model site effects), their ability to model heteroscedastic or non-Gaussian noise distributions and their suitability for use in a federated or decentralized learning environment. Gaussian process regression (GPR) was widely used in the beginning phases of normative modeling, which can flexibly model non-linear effects but does not computationally scale well when the training data increases (i.e., beyond a few thousand data points). In this work, we focus on Bayesian linear regression (BLR), which is highly scalable and flexible. For example, it can be combined with likelihood warping to model non-Gaussian effects. Hierarchical Bayesian regression (HBR) is another appealing choice as it has been used to better address multi-site datasets and allows for transfer learning (e.g., prediction for unseen sites) and can be estimated in a federated learning framework.

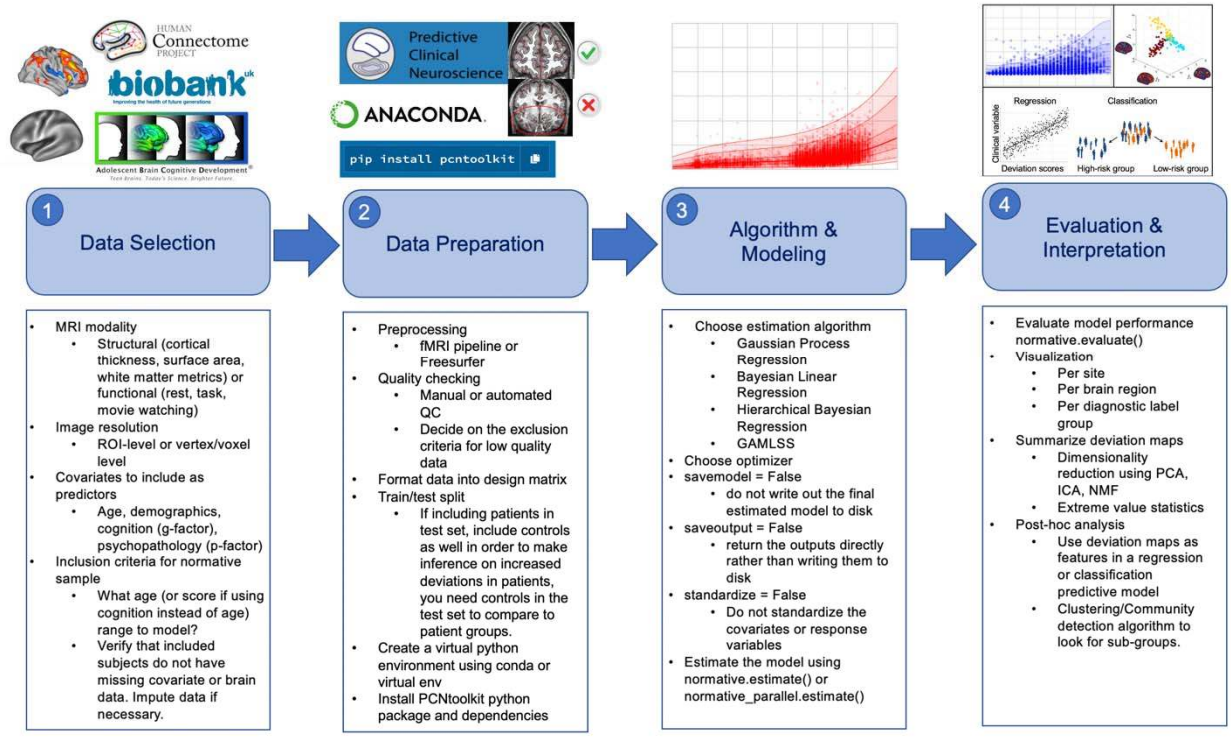

Figure 2 Practical Overview of Normative Modeling Framework. The workflow consists of four stages: data selection, data preparation, algorithm & modeling, and evaluation & interpretation, which are visualized by the numbered shaded blue boxes. The steps involved at each of these stages are summarized in the box below and highlighted in the images above.

Expertise needed to implement the protocol

We aimed to make this protocol user-friendly to the diverse community of neuroscience, including those with a non-technical background. The fundamental objective of this protocol is to learn how to implement the normative modeling framework via the PCNtoolkit software without being an expert in statistics and machine learning. You will be given enough knowledge to set up training and test sets, understand what data should be going into the model, interpret results, and make inferences on the results. Prerequisites of this protocol are basic familiarity with the Python programming language, a computer with WIFI, and a stable internet connection. Complete code, example data, and extensive documentation accompany this protocol; thus, writing code from scratch is unnecessary. Of course, it is our intention for readers to be inspired by this protocol and to use the normative modeling framework in new ways than presented here. If you wish to use the framework presented in this protocol beyond the provided code, familiarity with the Linux command line, bash scripting, setting up virtual environments, and submitting jobs to high-performance clusters would also be helpful.

Limitations

Big data requires automated QC

As datasets grow to meet the requirements of becoming population-level or big data, there is typically a need to rely on automated quality control metrics[63]. This means there is potential to unintentionally include poor quality data, which could, in turn, affect the results. The training and test dataset used in this protocol has been manually quality checked by visualizing every subject's raw T1w volume with their corresponding Freesurfer brain-mask as an overlay using an online (JavaScript-based) image viewer. Quality checking code and further instructions for use is made available on GitHub. These images were inspected for obvious quality issues, such as excess field-of-view cut-off, motion artifacts, or signal drop-out. Subjects that were flagged as having quality issues were excluded from the sample. Users should consider manually quality checking their own data if they wish to add on additional samples to the dataset.

Multi-site confounds

Pooling data from multiple sites is often a necessary step to create diverse datasets and reach sufficient sample sizes for machine learning analyses. When combining data from different studies, several challenges arise. First, there are often different MRI scanners at each site that also have different acquisition parameters. These MRI hardware and software divergences give

rise to substantial nuisance variance that must be properly accounted for when modeling the data. Second, there may be sampling differences, for example due to different inclusion criteria and definitions of diagnostic labels at each site. For example, one site may use the Structured Clinical Interview for DSM-5 (SCID-5) administered by a clinician or trained mental health professional who is familiar with the DSM-5 classification and diagnostic criteria, while another site may not have these resources in their study and therefore relies on self-report questionnaire data to define clinical labels. There is likely to be dissimilarities in the available demographic, cognitive, and clinical questionnaire data across sites as well which needs to be considered when deciding which studies to include in the training set. There is a careful balance that should be considered regarding the benefits gained from a new site joining the sample versus the nuisance site related variance that accompanies the addition of new sites.

Univariate nature

The PCNtoolkit can run models in parallel to speed up computation time; however, there is still a univariate nature, meaning a separate model is fit for each brain region. This univariate approach does not address the spatial autocorrelation[64–66] or functional heterogeneity (functional mis-registration) present in (f)MRI data[67]. This is an extra critical consideration when using functional MRI as the input. When using fMRI, you should consider using a hyperalignment algorithm (functional alignment)[68,69] to properly model the functional regions, as the spatial overlap of regions across individuals is not guaranteed with functional areas. Progress in addressing spatial autocorrelation in the context of normative modeling has been made[70], but modeling spatial autocorrelation is a difficult problem that requires further work.

Materials

Equipment

- Computing infrastructure: a Linux computer or HPC (SLURM or Torque) with enough space to store the imaging data of the train and test set.
    - If a Linux computer or server is unavailable, this protocol can also be run in Google Colab (for free). If using Google Colab, only a computer with an internet connection and modern internet browser (e.g., Chrome or Firefox) installed is necessary.
- Python installation.

- Recommended: Anaconda or virtual environment to manage the required python packages.

- PCNtoolkit python package version 0.20 (and dependencies) installed via pip.
- Covariates and response variables. Examples of these are provided with this tutorial.
  - Demographic and behavioral data used as predictor variables
    - Age, sex/gender, site/scanner ID, race/ethnicity, cognition, data quality metric (Euler number if structural, mean framewise displacement if functional)
  - Biological data to be predicted. An example structural MRI dataset is provided with this tutorial.
    - Structural MRI cortical thickness, surface area, subcortical volume
    - Functional MRI: parcellated task activation maps, resting-state networks

Procedure

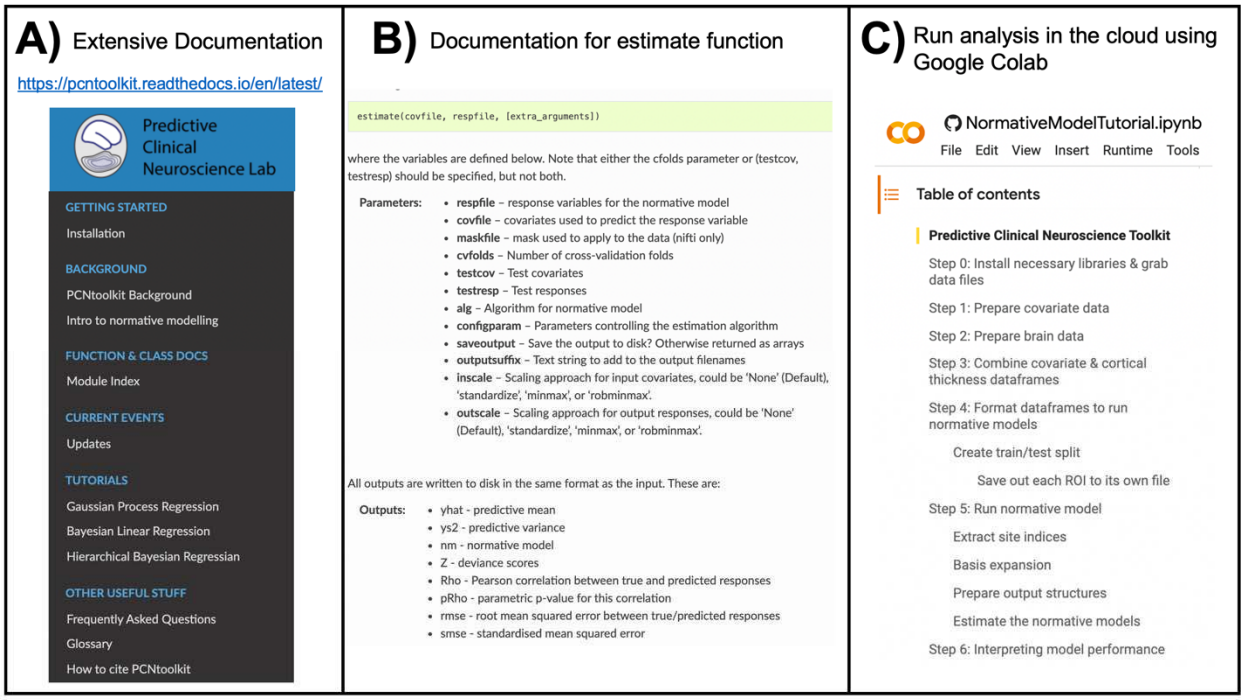

Figure 3 Overview of Resources for Running a Normative Modeling Analysis. A) Detailed documentation, including installation instructions, input/output descriptions of all classes and functions implemented in the python package, tutorials for all algorithms, frequently asked questions, a glossary explaining acronyms and other jargon, references to existing normative modeling literature, and a citation guide, is available online. **B)** Example of the documentation showing the required input, expected output of the main function used in the pcntoolkit software, the estimate function. **C)** All of the code and data used in this protocol is available to run in the

cloud via Google Colab. Additional tutorials (shown under the tutorials header in panel A) are also available to run in Google Colab.

Step 1: Install necessary packages and download the tutorial data

Timing = 1-3 minutes.

```
git clone https://github.com/predictive-clinical-neuroscience/PCNtoolkit-demo.git
# set this path to the git cloned PCNtoolkit-demo repository --> Uncomment whichever line you
need for either running on your own computer or on Google Colab.
#os.chdir('/Users/saigerutherford/repos/PCNtoolkit-demo/') # if running on your own computer, use
this line (change the path to match where you cloned the repository)
#os.chdir('PCNtoolkit-demo/') # if running on Google Colab, use this line
import os
pip install -r requirements.txt
```

Step 2: Prepare covariate data

Timing = 5-8 minutes.

For this tutorial we will use data from the Human Connectome Project Young Adult study, CAMCAN, and IXI to create a multi-site dataset. Our first step is to prepare and combine the covariate (age & sex) data from each site.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import joypy
from sklearn.model_selection import train_test_split
from pcntoolkit.normative import estimate, evaluate
from pcntoolkit.utils import create_bspline_basis, compute_MSLL
hcp = pd.read_csv('data/HCP1200_age_gender.csv')
cam = pd.read_csv('data/cam_age_gender.csv')
ixi = pd.read_csv('data/IXI_age_gender.csv')
cam_hcp = pd.merge(hcp, cam, how='outer')
cov = pd.merge(cam hcp, ixi, how='outer')
sns.set(font_scale=1.5, style='darkgrid')
sns.displot(cov, x="age", hue="site", multiple="stack", height=6)
cov.groupby(['site']).describe()
```

Step 3: Prepare brain data

Timing = 10-15 minutes.

Next, we will format and combine the MRI data. We are using cortical thickness maps that are created by running recon-all from Freesurfer (version 6.0). We need to merge the left and right hemisphere text files for each site, and then combine the different sites into a single dataframe. We reduce the dimensionality of our data by using ROIs from the Desikan-Killiany atlas.

```
cam = pd.read_csv('data/CAMCAN_aparc_thickness.csv')
hcpya = pd.read_csv('data/HCP1200_aparc_thickness.csv')
ixi = pd.read_csv('data/IXI_aparc_thickness.csv')
hcpya_cam = pd.merge(hcpya, cam, how='outer')
brain all = pd.merge(ixi, hcpya cam, how='outer')
```

We also want to include the Euler number as a covariate. So, we extracted the Euler number from each subject's recon-all output folder into a text file and we now need to format and combine these into our brain dataframe.

```
hcp_euler = pd.read_csv('data/hcp-ya_euler.csv')
cam_euler = pd.read_csv('data/cam_euler.csv')
ixi_euler = pd.read_csv('data/ixi_euler.csv')
hcp_euler['site'] = 'hcp'
cam_euler['site'] = 'cam'
ixi_euler['site'] = 'ixi'
hcp_euler.dropna(inplace=True)
cam_euler.dropna(inplace=True)
ixi_euler.dropna(inplace=True)
hcp_euler['rh_euler'] = hcp_euler['rh_euler'].astype(int)
hcp_euler['lh_euler'] = hcp_euler['lh_euler'].astype(int)
cam_euler['rh_euler'] = cam_euler['rh_euler'].astype(int)
cam_euler['lh_euler'] = cam_euler['lh_euler'].astype(int)
ixi_euler['rh_euler'] = ixi_euler['rh_euler'].astype(int)
ixi_euler['lh_euler'] = ixi_euler['lh_euler'].astype(int)
hcp_cam_euler = pd.merge(hcp_euler, cam_euler, how='outer')
df_euler = pd.merge(ixi_euler, hcp_cam_euler, how='outer')
```

We need to center the Euler number for each site. The Euler number is very site-specific so to use the same exclusion threshold across sites we need to center the site by subtracting the site median from all subjects at a site. Then we will take the square root and multiply by negative one and exclude any subjects with a square root above 10. If possible, your data should be visually inspected to verify that the data inclusion is not too strict or too lenient. Subjects above the Euler number threshold should be manually checked to verify and justify their exclusion due to poor data quality. This is just one approach for automated QC used by the developers of the PCNtoolkit. Other approaches such as the ENIGMA QC pipeline or UK Biobank's QC pipeline [63] are also viable options for automated QC.

```
df_euler['avg euler'] = df_euler[['lh euler','rh euler']].mean(axis=1)
df_euler.groupby(by='site').median()
df_euler['site median'] = df_euler['site']
df_euler['site_median'] = df_euler['site_median'].replace({'hcp':-43,'cam':-61,'ixi':-56})
df_euler['avg euler centered'] = df_euler['avg euler'] - df_euler['site median']
df_euler['avg_euler_centered_neg'] = df_euler['avg_euler_centered']*-1
df_euler['avg_euler_centered_neg_sqrt'] =
np.sqrt(np.absolute(df_euler['avg_euler_centered_neg']))
brain = pd.merge(df_euler, brain_all, how='inner')
brain_good = brain.query('avg euler centered neg sqrt < 10')
```

Step 4: Merge covariate & brain dataframes

Timing = 3-5 minutes.

Even though the normative modeling code needs the covariate and features (cortical thickness) in separate text files, we first need to merge them together to make sure that we have the same subjects in each file and that the rows (representing subjects) align.

```
# make sure to use how="inner" so that we only include subjects that have data in both the
covariate and the cortical thickness files
all_data = pd.merge(brain_good, cov, how='inner')
# Create a list of all the ROIs you want to run a normative model for
roi_ids = ['lh_MeanThickness_thickness',
           'rh_MeanThickness_thickness',
           'lh_bankssts_thickness',
           'lh_caudalanteriorcingulate_thickness',
           'lh_superiorfrontal_thickness',
           'rh superiorfrontal thickness']
```

Step 5: Format dataframe to run normative model

Timing = 3-5 minutes.

Exclude rows with NaN values and separate the brain features and covariates into their own dataframes.

```
from sklearn.model selection import train test split
all data = all data.dropna()

all_data_features = all_data[[subset=roi_ids]]
all data covariates = all data[['age','sex','site']]
```

Right now, the sites are coded in a single column using a string. We need to instead dummy encode the site variable so that there is a column for each site and the columns contain binary variables (0/1). The pandas package has a built-in function, pd.get_dummies to help us format the site column this way.

```
all_data_covariates = pd.get_dummies(all_data_covariates, columns=['site'])
all_data['Average_Thickness'] =
all data[['lh MeanThickness thickness','rh MeanThickness thickness']].mean(axis=1)
```

Take a sneak peek to see if there are any super obvious site effects. If there were, we would see a large separation in the fitted regression line for each site.

```
sns.set_theme(style="darkgrid",font_scale=1.5)
c = sns.lmplot(data=all_data, x="age", y="Average_Thickness", hue="site", height=6)
plt.ylim(1.5, 3.25)
plt.xlim(15, 95)
plt.show()
```

Step 6: Train/Test split

Timing = 3-5 minutes.

We will use 80% of the data for training and 20% for testing. We stratify our train/test split using the site variable to make sure that the train/test sets both contain data from all sites. The model wouldn't learn the site effects if all the data from one site was only in the test set. If your test set includes all patients, it is important to also include some controls (from the same site as patients) in the test set. To investigate the hypothesis that patients have more extreme deviation patterns

than controls, you need to verify that it is because they are patients not because they are in the test set, and you can check this by also including controls in the test set. In other words, you cannot separate site variation from diagnostic variation if you do not have control reference data.

```
X_train, X_test, y_train, y_test = train_test_split(all_data_covariates, all_data_features,
stratify=all data['site'], test size=0.2, random state=42)
```

Confirm that your train and test arrays are the same size (rows). You do not need the same size columns (subjects) in the train and test arrays, but the rows represent the covariate and responses which should be the same across train and test arrays.

```
tr cov size = X train.shape
tr_resp_size = y_train.shape
te_cov_size = X_test.shape
te_resp_size = y_test.shape
print("Train covariate size is: ", tr_cov_size)
print("Test covariate size is: ", te cov size)
print("Train response size is: ", tr_resp_size)
print("Test response size is: ", te_resp_size)
```

Save out each ROI to its own file. We set up the normative model so that for each response variable, **Y** (e.g. brain region) we fit a separate model. While the estimate function in the PCNtoolkit can handle having all the **Y**'s in a single text file, for this tutorial we are going to organize our **Y**'s so that they are each in their own text file and directory.

```
for c in y_train.columns:
      y_train[c].to_csv('resp_tr_' + c + '.txt', header=False, index=False)
      X_train.to_csv('cov_tr.txt', sep = '\t', header=False, index = False)
      y_train.to_csv('resp_tr.txt', sep = '\t', header=False, index = False)
for c in y_test.columns:
      y_test[c].to_csv('resp_te_' + c + '.txt', header=False, index=False)
      X_test.to_csv('cov_te.txt', sep = '\t', header=False, index = False)
      y_test.to_csv('resp_te.txt', sep = '\t', header=False, index = False)
! if [[ ! -e data/ROI_models/ ]]; then mkdir data/ROI_models; fi
! if [[ ! -e data/covariate_files/ ]]; then mkdir data/covariate_files; fi
! if [[ ! -e data/response_files/ ]]; then mkdir data/response_files; fi
! for i in `cat data/roi_dir_names`; do cd data/ROI_models; mkdir ${i}; cd ../../; cp
resp_tr_${i}.txt data/ROI_models/${i}/resp_tr.txt; cp resp_te_${i}.txt
data/ROI_models/${i}/resp_te.txt; cp cov_tr.txt data/ROI_models/${i}/cov_tr.txt; cp cov_te.txt
data/ROI_models/${i}/cov_te.txt; done
! mv resp_*.txt data/response_files/
! mv cov_t*.txt data/covariate_files/
```

Step 7: Run normative model

Timing = 1-2 minutes per model (multiply by number of ROIs/models).

```
# set this path to wherever your ROI_models folder is located (where you copied all of the
covariate & response text files to in Step 4)
data dir = '/Users/saigerutherford/repos/PCNToolkit-demo/data/ROI models/'
```

When we split the data into train and test sets, we did not reset the index. This means that the row numbers in the train/test matrices are still the same as before splitting the data. We will need

the test set row numbers of which subjects belong to which site to evaluate per site performance metrics, so we need to reset the row numbers in the train/test split matrices.

```
x col names = ['age', 'sex', 'site cam', 'site hcp', 'site ixi']
X_train = pd.read_csv('data/covariate_files/cov_tr.txt', sep='\t', header=None,
names=x col names)
X_test = pd.read_csv('data/covariate_files/cov_te.txt', sep='\t', header=None, names=x_col_names)
y train = pd.read csv('data/response files/resp tr.txt', sep='\t', header=None)
y_test = pd.read_csv('data/response_files/resp_te.txt', sep='\t', header=None)
X_train.reset_index(drop=True, inplace=True)
X_test.reset_index(drop=True, inplace=True)
y_train.reset_index(drop=True, inplace=True)
y test.reset index(drop=True, inplace=True)
```

Extract site indices so that we can evaluate the test metrics independently for each site.

```
cam_idx = X_test.index[X_test['site_cam' ]== 1].to_list()
hcp_idx = X_test.index[X_test['site_hcp'] == 1].to_list()
ixi_idx = X_test.index[X_test['site_ixi'] == 1].to_list()

# Save the site indices into a single list
sites = [cam_idx, hcp_idx, ixi_idx]

# Create a list with sites names to use in evaluating per-site metrics
site_names = ['cam', 'hcp', 'ixi']
```

Basis expansion

Now, we set up a B-spline basis set that allows us to perform nonlinear regression using a linear model. This basis is deliberately chosen to not to be too flexible so that in can only model relatively slowly varying trends. To increase the flexibility of the model you can change the parameterization (e.g., by adding knot points to the B-spline basis or increasing the order of the interpolating polynomial). Note that in the neuroimaging literature, it is more common to use a polynomial basis expansion for this. Piecewise polynomials like B-splines are superior to vanilla polynomial basis expansions because they do not introduce a global curvature. For further details on the use of B-splines see Fraza et al[19].

```
# Create a cubic B-spline basis (used for regression)
xmin = 10#16 # xmin & xmax are the boundaries for ages of participants in the dataset
xmax = 95#90
B = create_bspline_basis(xmin, xmax)
# create the basis expansion for the covariates for each of the
for roi in roi ids:
    print('Creating basis expansion for ROI:', roi)
    roi_dir = os.path.join(data_dir, roi)
    os.chdir(roi_dir)
    # create output dir
    os.makedirs(os.path.join(roi dir,'blr'), exist ok=True)
    # load train & test covariate data matrices
    X_tr = np.loadtxt(os.path.join(roi_dir, 'cov_tr.txt'))
    X_te = np.loadtxt(os.path.join(roi_dir, 'cov_te.txt'))
    # add intercept column
    X tr = np.concatenate((X tr, np.ones((X tr.shape[0],1))), axis=1)
    X_te = np.concatenate((X_te, np.ones((X_te.shape[0],1))), axis=1)
    np.savetxt(os.path.join(roi_dir, 'cov_int_tr.txt'), X_tr)
    np.savetxt(os.path.join(roi_dir, 'cov_int_te.txt'), X_te)

    # create Bspline basis set
    Phi = np.array([B(i) for i in X_tr[:,0]])
    Phis = np.array([B(i) for i in X_te[:,0]])
    X_tr = np.concatenate((X_tr, Phi), axis=1)
    X_te = np.concatenate((X_te, Phis), axis=1)
    np.savetxt(os.path.join(roi_dir, 'cov_bspline_tr.txt'), X_tr)
    np.savetxt(os.path.join(roi_dir, 'cov_bspline_te.txt'), X_te)
```

Prepare output structures

```
# Create pandas dataframes with header names to save out the overall and per-site model
evaluation metrics
blr_metrics = pd.DataFrame(columns = ['ROI', 'MSLL', 'EV', 'SMSE', 'RMSE', 'Rho'])
blr_site_metrics = pd.DataFrame(columns = ['ROI', 'site', 'y_mean', 'y_var', 'yhat_mean',
                                'yhat_var', 'MSLL', 'EV', 'SMSE', 'RMSE', 'Rho'])
```

Estimate the normative models

In this step, we estimate the normative models one at a time for each ROI. In principle, we could also do this on the whole data matrix at once (e.g., with the response variables stored in a n_subjects by n_brain_measures NumPy array or a text file). However, doing it this way gives us some extra flexibility in that it does not require that the subjects are the same for each of the brain measures. This code fragment will loop through each region of interest in the roi_ids list (set a few code blocks above) using Bayesian Linear Regression and evaluate the model on the independent test set. It will then compute error metrics such as the explained variance, mean standardized log-loss and Pearson correlation between true and predicted test responses separately for each scanning site. We supply the estimate function with a few specific arguments that are worthy of commenting on:

- alg = 'blr': specifies we should use Bayesian Linear Regression.

- optimizer = 'powell': use Powell's derivative-free optimization method (faster in this case than L-BFGS)
- savemodel = False: do not write out the final estimated model to disk
- saveoutput = False: return the outputs directly rather than writing them to disk
- standardize = False: Do not standardize the covariates or response variables

One important consideration is whether to re-scale or standardize the covariates or responses. Whilst this generally only has a minor effect on the final model accuracy, it has implications for the interpretation of models and how they are configured. If the covariates and responses are both standardized, the model will return standardized coefficients. If (as in this case) the response variables are not standardized, then the scaling both covariates and responses will be reflected in the estimated coefficients. Also, under the linear modelling approach employed here, if the coefficients are unstandardized and do not have a zero mean, it is necessary to add an intercept column to the design matrix. This is done in the code block above.

```
# Loop through ROIs
for roi in roi_ids:
    print('Running ROI:', roi)
    roi_dir = os.path.join(data_dir, roi)
    os.chdir(roi_dir)

    # configure the covariates to use. Change *_bspline_* to *_int_* to
    cov_file_tr = os.path.join(roi_dir, 'cov_bspline_tr.txt')
    cov_file_te = os.path.join(roi_dir, 'cov_bspline_te.txt')

    # load train & test response files
    resp_file_tr = os.path.join(roi_dir, 'resp_tr.txt')
    resp_file_te = os.path.join(roi_dir, 'resp_te.txt')

    # run a basic model
    yhat_te, s2_te, nm, Z, metrics_te = estimate(cov_file_tr,
                                                 resp_file_tr,
                                                 testresp=resp_file_te,
                                                 testcov=cov_file_te,
                                                 alg = 'blr',
                                                 optimizer = 'powell',
                                                 savemodel = False,
                                                 saveoutput = False,
                                                 standardize = False)
    # display and save metrics
    print('EV=', metrics_te['EXPV'][0])
    print('RHO=', metrics_te['Rho'][0])
    print('MSLL=', metrics_te['MSLL'][0])
    blr_metrics.loc[len(blr_metrics)] = [roi, metrics_te['MSLL'][0],
    metrics_te['EXPV'][0], metrics_te['SMSE'][0], metrics_te['RMSE'][0],
    metrics_te['Rho'][0]]

    # Compute metrics per site in test set, save to pandas df
    # load true test data
    X_te = np.loadtxt(cov_file_te)
    y_te = np.loadtxt(resp_file_te)
    y_te = y_te[:, np.newaxis] # make sure it is a 2-d array

    # load training data (required to compute the MSLL)
    y_tr = np.loadtxt(resp_file_tr)
    y_tr = y_tr[:, np.newaxis]

    for num, site in enumerate(sites):
        y_mean_te_site = np.array([[np.mean(y_te[site])]])
        y_var_te_site = np.array([[np.var(y_te[site])]])
        yhat_mean_te_site = np.array([[np.mean(yhat_te[site])]])
        yhat_var_te_site = np.array([[np.var(yhat_te[site])]])

        metrics_te_site = evaluate(y_te[site], yhat_te[site], s2_te[site],
        y_mean_te_site, y_var_te_site)

        site_name = site_names[num]
        blr_site_metrics.loc[len(blr_site_metrics)] = [roi, site_names[num],
                                                       y_mean_te_site[0],
                                                       y_var_te_site[0],
                                                       yhat_mean_te_site[0],
                                                       yhat_var_te_site[0],
                                                       metrics_te_site['MSLL'][0],
                                                       metrics_te_site['EXPV'][0],
                                                       metrics_te_site['SMSE'][0],
                                                       metrics_te_site['RMSE'][0],
                                                       metrics_te_site['Rho'][0]]
```

Anticipated Results

There are multiple end products created from running a normative model analysis. First, the evaluation metrics for each model (brain region) are saved to a file. In this tutorial, we saved the metrics to a CSV file format, however, in the pcn.estimate() function you could set the argument 'binary = True' which would save the metrics in pickle (.pkl) format. Pickle format is good to use if you are estimating many models in parallel on a large dataset, as it is faster because it avoids reading/writing intermediate text files. These metrics are further summarized into per site metrics to check model fit for each site included in the test set. The short and full names of the evaluation metrics and a brief interpretation guide is summarized below in Box 1. The evaluation metrics can be visualized in numerous formats, histograms/density plots, scatter plots with fitted centiles, or brain-space visualizations. Several examples of these visualizations are shown in Figure 4. Quality checking the normative model evaluation metrics should be done to ensure proper model estimation. If a model fits well to the data, the evaluation metrics should follow a Gaussian distribution. The model estimation step should properly handle confounding site effects, nevertheless, it is also a good idea to check per site metrics to make sure the model is fitting all sites equally well and that there are no obvious site outliers.

Box 1: Normative Model Evaluation Metrics

Variable name	Full name	Definition	Interpretation
y_d	True data		
$\hat{y}_d$	Predictive mean		
σ_d^2	Predictive noise variance	Represents uncertainty in the data.	
$(\sigma_*^2)_d$	Predictive modeling variance	Represents uncertainty in model estimation.	
Z	Deviation score	A statistical estimate (Z-score) of how much each subject deviates from the normative range.	Z > 2 'extreme positive deviation' Z < -2 'extreme negative deviation'
Rho	Pearson correlation	A measure of linear correlation between true and	Ranges between -1 and 1. Closer to 1 = better model

	between true and predicted responses	predicted responses. It is the ratio between the covariance of true and predicted values and the product of their standard deviations.	performance.
pRho	Parametric p-value for the Pearson correlation	The probability of obtaining test results at least as extreme as the results actually observed, under the assumption that the null hypothesis is true.	Ranges between 0 and 1. Closer to 0 = more statistically significant.
SMSE	Standardized mean squared error	The square root of the squared residual between the mean prediction and the target at each test point, averaged over samples in the test set, normalized by the variance of the targets in the test set.	Closer to 0 = better (more accurate) model performance.
EV	Explained variance	The proportion to which the predicted value accounts for the variance of the true value. Sensitive to the mean fit, dependent on flexibility of the model.	Closer to 1 = better model performance.
MSLL	Mean standardized log-loss	The log loss minus the loss that would be obtained under the trivial model which predicts using a Gaussian with the mean and variance of the training data, averaged over the test set. Sensitive to the variance, penalizes the flexibility of the model.	More negative = better model performance.

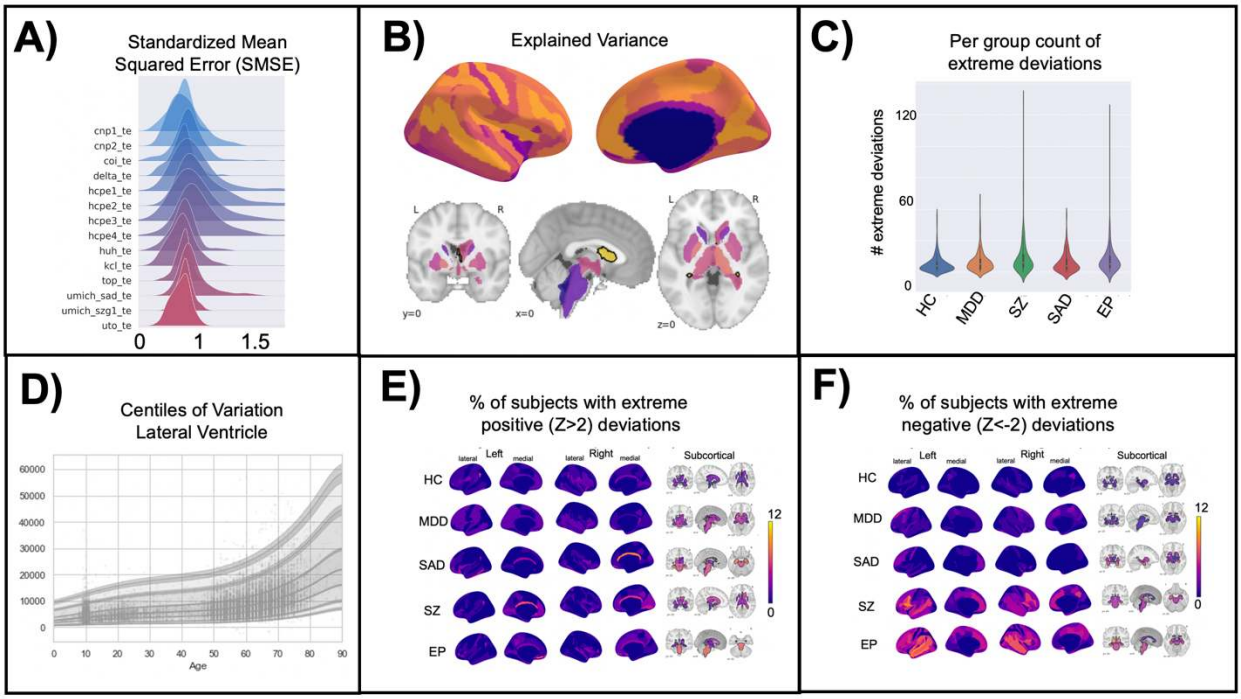

Figure 4 Visualization of Normative Model Evaluation Metrics. A) A ridge plot showing the distribution across all brain regions of the standardized mean squared error (SMSE), an evaluation metric that represents accuracy, visualized for each site in the test set. Visualizing for each test site can help identify if there are sites where the model is performing poorly. Ideally, the distribution will be Gaussian and should look similar across all sites. Small shifts in the mean across sites is to be expected and is acceptable. **B)** Explained variance is shown for cortical thickness of every brain region in the Destrieux parcellation) and volume of subcortical regions. Visualizing the evaluation metrics in brain space helps to identify patterns and see the big picture. **C)** The number of extreme deviations (both positive and negative) are counted for each individual in the test set, group ID is used to plot the distribution of the extreme deviation count for each group. A statistical test can be done on the count to determine if there is a significant difference between groups. Testing group differences in the count of deviations does not require there to be spatial overlap of the deviations within the group (i.e., this test can account for within-group heterogeneity of deviations). **D)** The normative trajectory for an example brain region (lateral ventricle) showing age (x-axis) versus the predicted volume (y-axis). The centiles of variation are shown by the lines and shaded confidence intervals. Each subject in the test set is plotted as a single point. **E-F)** Extreme deviations, separated into positive (**E**) and negative (**F**), are summarized for each group. For each brain region, the number of subjects with an extreme deviation in that region is counted, then divided by the group sample size, to show the percent of subjects with an extreme deviation. These visualizations demonstrate the benefit of normative modeling as there is within group heterogeneity that other methods (i.e., case-control group difference testing) are not equipped to handle. Abbreviations: HC = Controls, MDD=Major Depressive Disorder, SZ=Schizophrenia, SAD=Social Anxiety Disorder, EP=Early Psychosis.

There are many interesting analyses that can be conducted using the outputs of normative modeling (deviation scores). A full tutorial on each of these analyses is outside the scope of this

protocol. However, on GitHub, we include code examples (python notebooks that can be run via Colab) of the following post-hoc analysis:

- Using deviation scores as predictors in a regression and classification.
- Dimensionality reduction to get a latent representation of deviation scores.
- Classical case-control testing (univariate t-tests) on deviation maps compared to univariate t-tests on the true data.

A benefit of the PCNtoolkit software for normative modeling, that sets our approach apart from other implementations[71], is the fine-scale resolution allowed by the model. Other normative modeling work[71] has focused on modeling gross features such as total brain volume or gray matter volume, which is not adequate for normative modeling applied to mental health conditions and neurodevelopmental disorders, where the effects are subtle and widespread (individuals within a patient group tend to deviate in different regions, see Figure 4E-F) across the cortex and subcortex and averaging over large brain areas usually overlook these elusive psychiatric effects.

Troubleshooting

We re-iterate that there is additional documentation available online through read the docs including additional tutorials for other algorithm implementations (Gaussian Process Regression and Hierarchical Bayesian Regression), a glossary to clarify the jargon associated with the software, a reference guide with links to normative modeling publications, and a frequently asked questions page where many common errors (and their solutions) are discussed in detail.

The problems encountered when troubleshooting a normative modeling analysis can fall into three categories: computing errors, data issues, and misunderstanding or misinterpreting the outputs. The computing errors might involve python or the computer hardware. Potential python errors may include installation of python or installation of the necessary packages and their dependencies. We recommend using Anaconda to install python 3.8 (required for this tutorial) on your system, and the use of a virtual environment for the PCNtoolkit to ensure that the packages required for normative modeling do not interfere with other python versions and packages you may have installed on your system. In general, it is good to have a virtual environment setup for each project or analysis. If you are unfamiliar with setting up virtual environments, and run into issues with python, it is always an option to run the analysis in the cloud via Colab which

eliminates the need to setup python on your own system. Hardware problems might include lack of memory to store the data or models running very slowly due to outdated hardware. These hardware errors do not have an easy solution, and we recommend using Google Colab to run normative modeling analysis if your personal computer or server is very slow or lacks the storage space. Data issues that may be encountered are data missing not at random (see Experimental Design section re: caution using data imputation), improperly coded data (i.e., strings instead of integers or floats, NaN values coded incorrectly), collinearity of columns in the covariate design matrix, or outlier data that does not make biological sense (i.e., negative cortical thickness values, negative age values). While these data error can be incredibly frustrating to troubleshoot, they can typically be fixed by careful quality checking of the input data and removal of bad ROIs or subjects as needed. Finally, an example of interpretation confusion may be poor model performance on a certain brain region or site. This can usually be addressed by returning to the input data for additional quality checking.

Data Availability

All data and code are available on GitHub in the format of a jupyter python notebook that can be run in the cloud (for free) using Google Colab. Examples of post-hoc analysis and code for visualizing the evaluation metrics can also be found on GitHub.

Acknowledgements

This research was supported by grants from the European Research Council (ERC, grant "MENTALPRECISION" 10100118 and "BRAINMINT" 802998), the Wellcome Trust under an Innovator award ("BRAINCHART", 215698/Z/19/Z) and a Strategic Award (098369/Z/12/Z), the Dutch Organisation for Scientific Research (VIDI grant 016.156.415). TW also gratefully acknowledges the Niels Stensen Fellowship as well as the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie Grant agreement No. 895011.

Conflict of Interests

CFB is director and shareholder of SBGNeuro Ltd. HGR received speaker's honorarium from Lundbeck and Janssen. The other authors report no conflicts of interest.

References

1. Wang, D. *et al.* Parcellating cortical functional networks in individuals. *Nat. Neurosci.* **18**, 1853–1860 (2015).
2. Finn, E. S. & Todd Constable, R. Individual variation in functional brain connectivity: implications for personalized approaches to psychiatric disease. *Dialogues Clin. Neurosci.* **18**, 277–287 (2016).
3. Braga, R. M. & Buckner, R. L. Parallel Interdigitated Distributed Networks within the Individual Estimated by Intrinsic Functional Connectivity. *Neuron* **95**, 457-471.e5 (2017).
4. Poldrack, R. A. Precision Neuroscience: Dense Sampling of Individual Brains. *Neuron* **95**, 727–729 (2017).
5. Vanderwal, T. *et al.* Individual differences in functional connectivity during naturalistic viewing conditions. *NeuroImage* **157**, 521–530 (2017).
6. Braun, U. *et al.* From Maps to Multi-dimensional Network Mechanisms of Mental Disorders. *Neuron* **97**, 14–31 (2018).
7. Gratton, C. *et al.* Defining Individual-Specific Functional Neuroanatomy for Precision Psychiatry. *Biol. Psychiatry* **88**, 28–39 (2020).
8. Hyman, S. E. Can neuroscience be integrated into the DSM-V? *Nat. Rev. Neurosci.* **8**, 725–732 (2007).
9. Insel, T. *et al.* Research Domain Criteria (RDoC): Toward a New Classification Framework for Research on Mental Disorders. *Am. J. Psychiatry* **167**, 748–751 (2010).
10. Michelini, G., Palumbo, I. M., DeYoung, C. G., Latzman, R. D. & Kotov, R. Linking RDoC and HiTOP: A new interface for advancing psychiatric nosology and neuroscience. *Clin. Psychol. Rev.* **86**, 102025 (2021).

11. Narrow, W. E. & Kuhl, E. A. Dimensional approaches to psychiatric diagnosis in DSM-5. *J. Ment. Health Policy Econ.* **14**, 197–200 (2011).

12. Feczko, E. *et al.* The Heterogeneity Problem: Approaches to Identify Psychiatric Subtypes. *Trends Cogn. Sci.* **23**, 584–601 (2019).

13. Shen, X. *et al.* Using connectome-based predictive modeling to predict individual behavior from brain connectivity. *Nat. Protoc.* **12**, 506–518 (2017).

14. Sripada, C. *et al.* Basic Units of Inter-Individual Variation in Resting State Connectomes. *Sci. Rep.* **9**, 1900 (2019).

15. Woo, C.-W., Chang, L. J., Lindquist, M. A. & Wager, T. D. Building better biomarkers: brain models in translational neuroimaging. *Nat. Neurosci.* **20**, 365–377 (2017).

16. Marquand, A. F. *et al.* Conceptualizing mental disorders as deviations from normative functioning. *Mol. Psychiatry* **24**, 1415–1424 (2019).

17. Gau, R. *et al.* Brainhack: Developing a culture of open, inclusive, community-driven neuroscience. *Neuron* **109**, 1769–1775 (2021).

18. Olah, C. & Carter, S. Research Debt. *Distill* **2**, e5 (2017).

19. Fraza, C. J., Dinga, R., Beckmann, C. F. & Marquand, A. F. Warped Bayesian Linear Regression for Normative Modelling of Big Data. *bioRxiv* 2021.04.05.438429 (2021) doi:10.1101/2021.04.05.438429.

20. Dinga, R. *et al. Normative modeling of neuroimaging data using generalized additive models of location scale and shape*. http://biorxiv.org/lookup/doi/10.1101/2021.06.14.448106 (2021) doi:10.1101/2021.06.14.448106.

21. Kia, S. M. *et al.* Hierarchical Bayesian Regression for Multi-site Normative Modeling of Neuroimaging Data. in *Medical Image Computing and Computer Assisted Intervention –*

MICCAI 2020 (eds. Martel, A. L. et al.) 699–709 (Springer International Publishing, 2020). doi:10.1007/978-3-030-59728-3_68.

22. Kia, S. M. *et al.* Federated Multi-Site Normative Modeling using Hierarchical Bayesian Regression. *bioRxiv* 2021.05.28.446120 (2021) doi:10.1101/2021.05.28.446120.
23. Floris, D. L. *et al.* Atypical Brain Asymmetry in Autism—A Candidate for Clinically Meaningful Stratification. *Biol. Psychiatry Cogn. Neurosci. Neuroimaging* (2020) doi:10.1016/j.bpsc.2020.08.008.
24. Zabihi, M. *et al.* Dissecting the Heterogeneous Cortical Anatomy of Autism Spectrum Disorder Using Normative Models. *Biol. Psychiatry Cogn. Neurosci. Neuroimaging* **4**, 567–578 (2019).
25. Zabihi, M. *et al.* Fractionating autism based on neuroanatomical normative modeling. *Transl. Psychiatry* **10**, 1–10 (2020).
26. Wolfers, T. *et al.* Individual differences v. the average patient: mapping the heterogeneity in ADHD using normative models. *Psychol. Med.* **50**, 314–323 (2020).
27. Wolfers, T. *et al.* Refinement by integration: aggregated effects of multimodal imaging markers on adult ADHD. *J. Psychiatry Neurosci. JPN* **42**, 386–394 (2017).
28. Verdi, S., Marquand, A. F., Schott, J. M. & Cole, J. H. Beyond the average patient: how neuroimaging models can address heterogeneity in dementia. *Brain* (2021) doi:10.1093/brain/awab165.
29. Wolfers, T. *et al. Extensive brain structural heterogeneity in individuals with schizophrenia and bipolar disorder*. http://medrxiv.org/lookup/doi/10.1101/2020.05.08.20095091 (2020) doi:10.1101/2020.05.08.20095091.

30. Wolfers, T. *et al.* Mapping the Heterogeneous Phenotype of Schizophrenia and Bipolar Disorder Using Normative Models. *JAMA Psychiatry* **75**, 1146–1155 (2018).

31. Wolfers, T. *et al.* Replicating extensive brain structural heterogeneity in individuals with schizophrenia and bipolar disorder. *Hum. Brain Mapp.* **42**, 2546–2555 (2021).

32. Sripada, C., Angstadt, M., Rutherford, S. & Taxali, A. Brain Network Mechanisms of General Intelligence. *bioRxiv* 657205 (2019) doi:10.1101/657205.

33. Sripada, C. *et al. Brain Connectivity Patterns in Children Linked to Neurocognitive Abilities*. http://biorxiv.org/lookup/doi/10.1101/2020.09.10.291500 (2020) doi:10.1101/2020.09.10.291500.

34. Rosenberg, M. D. *et al.* A neuromarker of sustained attention from whole-brain functional connectivity. *Nat. Neurosci.* **19**, 165–171 (2015).

35. Rosenberg, M. D. *et al.* Functional connectivity predicts changes in attention observed across minutes, days, and months. *Proc. Natl. Acad. Sci.* **117**, 3797–3807 (2020).

36. Marquand, A. F., Haak, K. V. & Beckmann, C. F. Functional corticostriatal connection topographies predict goal directed behaviour in humans. *Nat. Hum. Behav.* **1**, (2017).

37. Marquand, A. *et al.* Quantitative prediction of subjective pain intensity from whole-brain fMRI data using Gaussian processes. *NeuroImage* **49**, 2178–2189 (2010).

38. Wager, T. D. *et al.* An fMRI-Based Neurologic Signature of Physical Pain. *N. Engl. J. Med.* **368**, 1388–1397 (2013).

39. Sripada, C., Angstadt, M., Rutherford, S., Taxali, A. & Shedden, K. Toward a "treadmill test" for cognition: Improved prediction of general cognitive ability from the task activated brain. *Hum. Brain Mapp.* **41**, 3186–3197 (2020).

40. Sripada, C., Taxali, A., Angstadt, M. & Rutherford, S. *Boost in Test-Retest Reliability in Resting State fMRI with Predictive Modeling*. http://biorxiv.org/lookup/doi/10.1101/796714 (2019) doi:10.1101/796714.

41. Finn, E. S. *et al.* Functional connectome fingerprinting: identifying individuals using patterns of brain connectivity. *Nat. Neurosci.* **18**, 1664–1671 (2015).

42. Wang, H.-T. *et al.* Finding the needle in a high-dimensional haystack: Canonical correlation analysis for neuroscientists. *NeuroImage* **216**, 116745 (2020).

43. Smith, S. M. *et al.* A positive-negative mode of population covariation links brain connectivity, demographics and behavior. *Nat. Neurosci.* **18**, 1565–1567 (2015).

44. Dadi, K. *et al.* Benchmarking functional connectome-based predictive models for resting-state fMRI. *NeuroImage* **192**, 115–134 (2019).

45. Lake, E. M. R. *et al.* The Functional Brain Organization of an Individual Allows Prediction of Measures of Social Abilities Transdiagnostically in Autism and Attention-Deficit/Hyperactivity Disorder. *Biol. Psychiatry* **86**, 315–326 (2019).

46. Cole, J. H. & Franke, K. Predicting Age Using Neuroimaging: Innovative Brain Ageing Biomarkers. *Trends Neurosci.* **40**, 681–690 (2017).

47. Han, L. K. M. *et al.* Brain aging in major depressive disorder: results from the ENIGMA major depressive disorder working group. *Mol. Psychiatry* 1–16 (2020) doi:10.1038/s41380-020-0754-0.

48. Sturmfels, P. *et al.* A Domain Guided CNN Architecture for Predicting Age from Structural Brain Images. *ArXiv180804362 Cs Stat* (2018).

49. Donders, A. R. T., van der Heijden, G. J. M. G., Stijnen, T. & Moons, K. G. M. Review: A gentle introduction to imputation of missing values. *J. Clin. Epidemiol.* **59**, 1087–1091 (2006).

50. Madley-Dowd, P., Hughes, R., Tilling, K. & Heron, J. The proportion of missing data should not be used to guide decisions on multiple imputation. *J. Clin. Epidemiol.* **110**, 63–73 (2019).

51. Casey, B. J. *et al.* The Adolescent Brain Cognitive Development (ABCD) study: Imaging acquisition across 21 sites. *Dev. Cogn. Neurosci.* **32**, 43–54 (2018).

52. Thompson, P. M. *et al.* ENIGMA and global neuroscience: A decade of large-scale studies of the brain in health and disease across more than 40 countries. *Transl. Psychiatry* **10**, 100 (2020).

53. Beer, J. C. *et al.* Longitudinal ComBat: A method for harmonizing longitudinal multi-scanner imaging data. *NeuroImage* **220**, 117129 (2020).

54. Fortin, J.-P. *et al.* Harmonization of multi-site diffusion tensor imaging data. *NeuroImage* **161**, 149–170 (2017).

55. Fortin, J.-P. *et al.* Harmonization of cortical thickness measurements across scanners and sites. *NeuroImage* **167**, 104–120 (2018).

56. Johnson, W. E., Li, C. & Rabinovic, A. Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics* **8**, 118–127 (2007).

57. Nygaard, V., Rødland, E. A. & Hovig, E. Methods that remove batch effects while retaining group differences may lead to exaggerated confidence in downstream analyses. *Biostatistics* **17**, 29–39 (2016).

58. Marquand, A. F., Wolfers, T., Mennes, M., Buitelaar, J. & Beckmann, C. F. Beyond Lumping and Splitting: A Review of Computational Approaches for Stratifying Psychiatric Disorders. *Biol. Psychiatry Cogn. Neurosci. Neuroimaging* **1**, 433–447 (2016).

59. Huertas, I. *et al.* A Bayesian spatial model for neuroimaging data based on biologically informed basis functions. *NeuroImage* **161**, 134–148 (2017).

60. Kia, S. M. & Marquand, A. Normative Modeling of Neuroimaging Data using Scalable Multi-Task Gaussian Processes. *ArXiv180601047 Cs Stat* (2018).

61. Rahimi, A. & Recht, B. Random Features for Large-Scale Kernel Machines. 8.

62. Lv, J. *et al.* Individual deviations from normative models of brain structure in a large cross-sectional schizophrenia cohort. *Mol. Psychiatry* 1–12 (2020) doi:10.1038/s41380-020-00882-5.

63. Alfaro-Almagro, F. *et al.* Image processing and Quality Control for the first 10,000 brain imaging datasets from UK Biobank. *NeuroImage* **166**, 400–424 (2018).

64. Burt, J. B., Helmer, M., Shinn, M., Anticevic, A. & Murray, J. D. Generative modeling of brain maps with spatial autocorrelation. *NeuroImage* **220**, 117038 (2020).

65. Shinn, M. *et al.* Spatial and temporal autocorrelation weave human brain networks. *bioRxiv* 2021.06.01.446561 (2021) doi:10.1101/2021.06.01.446561.

66. Smith, S. M. & Nichols, T. E. Threshold-free cluster enhancement: addressing problems of smoothing, threshold dependence and localisation in cluster inference. *Neuroimage* **44**, 83–98 (2009).

67. Bijsterbosch, J. *et al.* Challenges and future directions for representations of functional brain organization. *Nat. Neurosci.* **23**, 1484–1495 (2020).

68. Haxby, J. V., Guntupalli, J. S., Nastase, S. A. & Feilong, M. Hyperalignment: Modeling shared information encoded in idiosyncratic cortical topographies. *eLife* **9**, e56601 (2020).

69. Busch, E. L. *et al.* Hybrid hyperalignment: A single high-dimensional model of shared information embedded in cortical patterns of response and functional connectivity. *NeuroImage* **233**, 117975 (2021).

70. Kia, S. M., Beckmann, C. F. & Marquand, A. F. Scalable Multi-Task Gaussian Process Tensor Regression for Normative Modeling of Structured Variation in Neuroimaging Data. *ArXiv180800036 Cs Stat* (2018).

71. Bethlehem, R. a. I. *et al.* Brain charts for the human lifespan. *bioRxiv* 2021.06.08.447489 (2021) doi:10.1101/2021.06.08.447489.