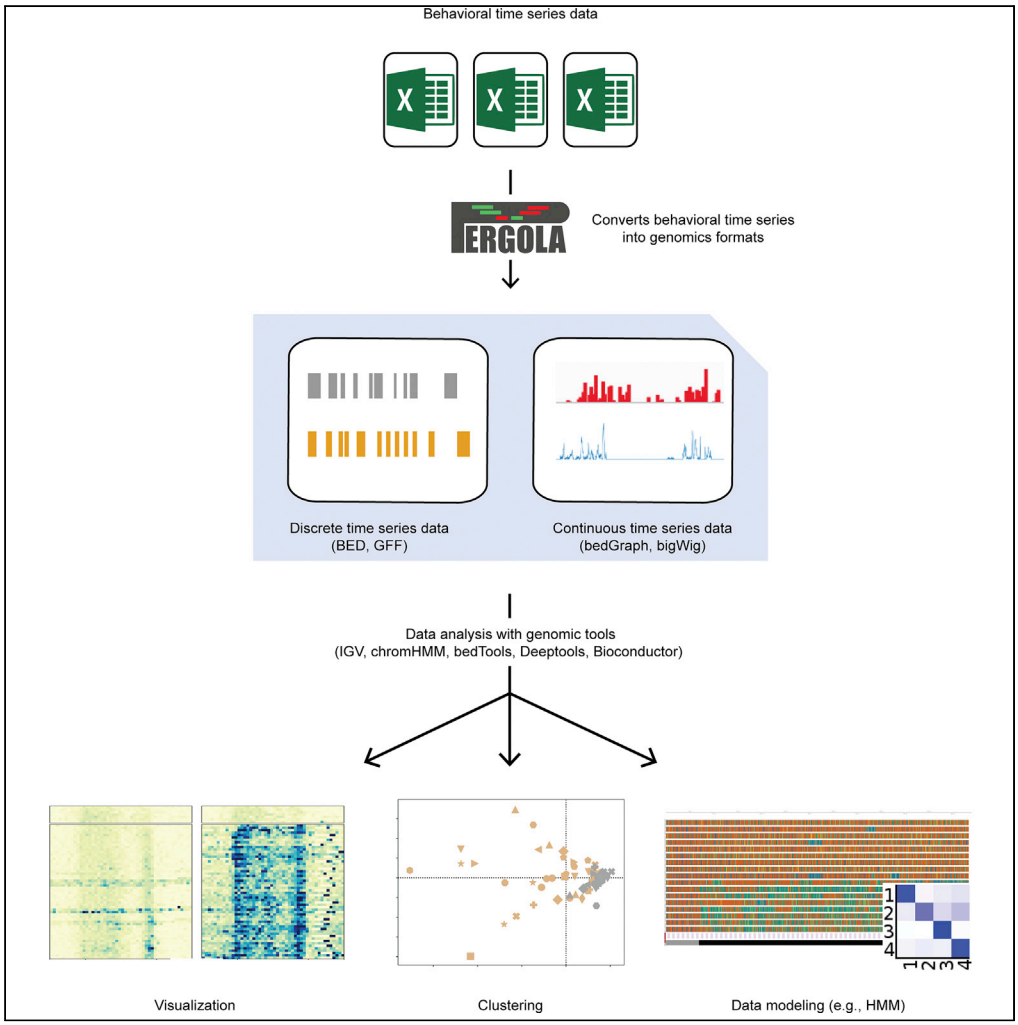
iScience

CellPress

Article

Pergola: Boosting Visualization and Analysis of Longitudinal Data by Unlocking Genomic Analysis Tools



Jose Espinosa-Carrasco, Ionas Erb, Toni Hermoso Pulido, Julia Ponomarenko, Mara Dierssen, Cedric Notredame

mara.dierssen@crg.eu (M.D.)
cedric.notredame@crg.eu (C.N.)

HIGHLIGHTS

Genomic formats can be repurposed to handle behavioral data without information loss

Pergola can store, visualize, and analyze behavioral data with genomics tools

Genomic formats and tools ease standardized, interoperable, and reproducible analysis

Three existing sets of analysis on mouse, fly, and worms are reproduced in this study

Espinosa-Carrasco et al., iScience 9, 244–257 November 30, 2018
https://doi.org/10.1016/j.isci.2018.10.023

iScience

CellPress

Article

Pergola: Boosting Visualization and Analysis of Longitudinal Data by Unlocking Genomic Analysis Tools

Jose Espinosa-Carrasco,[1,2] Ionas Erb,[1] Toni Hermoso Pulido,[1] Julia Ponomarenko,[1,3] Mara Dierssen,[1,3,4,*] and Cedric Notredame[1,3,5,*]

SUMMARY

The growing appetite of behavioral neuroscience for automated data production is prompting the need for new computational standards allowing improved interoperability, reproducibility, and shareability. We show here how these issues can be solved by repurposing existing genomic formats whose structure perfectly supports the handling of time series. This allows existing genomic analysis and visualization tools to be deployed onto behavioral data. As a proof of principle, we implemented the conversion procedure in Pergola, an open source software, and used genomics tools to reproduce results obtained in mouse, fly, and worm. We also show how common genomics techniques such as principal component analysis, hidden Markov modeling, and volcano plots can be deployed on the reformatted behavioral data. These analyses are easy to share because they depend on the scripting of public software. They are also easy to reproduce thanks to their integration within Nextflow, a workflow manager using containerized software.

INTRODUCTION

Fine-grained longitudinal recordings are one of the fastest growing collections of biological and societal data (de Montjoye et al., 2015; Jensen et al., 2014). Behavior is a prime target for these types of measurements (Price et al., 2017) since it constitutes a high-level phenotype that links genetics, development, neurobiology, evolution, environment, and social influences (Gomez-Marin et al., 2014). Novel high-throughput platforms for monitoring behavior have recently enabled unprecedented amounts of data to be collected (Schaefer and Claridge-Chang, 2012). However, these data present novel challenges, and their effective comparison and reproducible analysis across different systems and platforms will require improved interoperability (Anderson and Perona, 2014; Munafò et al., 2017). Here, we show how standards established for genomics can be repurposed to handle behavioral data in a fully interoperable fashion. This simple solution made it possible to analyze behavioral neuroscience datasets gathered on three model systems (Espinosa-Carrasco et al., 2018; Robie et al., 2017; Yemini et al., 2013) using standard genomic formats and software tools.

High-throughput behavior-monitoring platforms can be classified into two categories: sensor- and video-based systems. Sensor-based systems measure selected parameters such as vocalizations, activity, feeding, and oxygen consumption. For mice, such devices include PHECOMP (Espinosa-Carrasco et al., 2018), PhenoMaster (Suez et al., 2014) and CLAMS (Solt et al., 2012). Video-based systems use computer vision techniques to track movements. Optimized video setups have been developed for the main model organisms, including mouse (Hong et al., 2015), worm (Yemini et al., 2013), fly (Robie et al., 2017), and zebrafish (Pérez-Escudero et al., 2014). Even though the recorded signals are different across systems, the outputs are comparable in the sense that they consist of discrete or continuous time series of behavioral quantitative or qualitative readouts. These time series are usually processed using commercial software shipped with the platform, often in combination with ad hoc implemented analyses. Although this approach is entirely suitable for establishing key biological results, the lack of common standards for longitudinal data processing hampers the ability to share established computational analysis procedures and data (Wilkinson et al., 2016), thus potentially limiting comparisons and reproducibility across different systems. Improving interoperability is not, however, an easy task, because it requires the whole community to agree on common standards, formats, and procedures. In genomics, the establishment of such standards has taken about 15 years and work is still in progress (Field et al., 2011; Karsch-Mizrachi et al., 2018). In this

[1]Centre for Genomic Regulation (CRG), The Barcelona Institute of Science and Technology, Dr. Aiguader 88, Barcelona 08003, Spain

[2]Institute for Research in Biomedicine (IRB Barcelona), The Barcelona Institute of Science and Technology, Baldiri Reixac, 10, Barcelona 08028, Spain

[3]Universitat Pompeu Fabra (UPF), Barcelona, Spain

[4]Centro de Investigación Biomédica en Red de Enfermedades Raras (CIBERER), Valencia, Spain

[5]Lead Contact

*Correspondence: mara.dierssen@crg.eu (M.D.), cedric.notredame@crg.eu (C.N.)

https://doi.org/10.1016/j.isci.2018.10.023

A Sample file

id	event_start	event_end	type_of_event	value
1	1335959790	1335959858	sc	0.06
1	1335961018	1335961107	sc	0.04
1	1335963462	1335963645	sc	0.02

B BED file

chr1	0	68	sc	0.06	+	0	68	255,180,0
chr1	1228	1317	sc	0.04	+	1228	1317	255,180,0
chr1	3702	3855	sc	0.02	+	3702	3855	255,220,0

C Data visualization

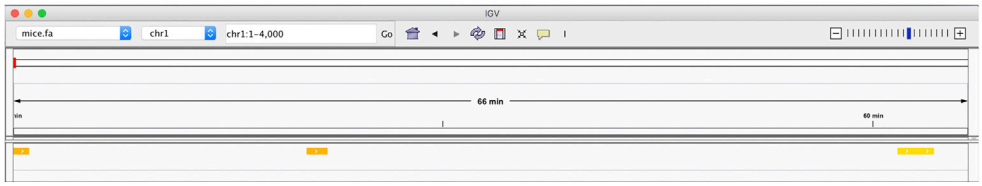


Figure 1. Formatting of Common Behavioral Recordings into BED as an Example of the Pergola Approach to Format Data

(A) Typical tabulated (CSV, TSV, or XLSX) behavioral recording with three events (lines 2–4) being coded with an animal tag (id column), start (event_start column) and end time (event_end column), a typology (type_of_event), and a quantitative value (value column).

(B) Once mapped onto the Pergola generic ontology, these column labels are translated into a BED file format where the nucleotide relative index positions (columns 2 and 3) are used to specify the start and termination of the event whose nature is coded in the attribute column (#4) and quantified in the quantification column (#6). The BED format also supports specific coloring for the considered event (RGB values in the last column). A single BED file will contain all the events of an individual animal ID. Pergola will generate a FASTA file name chromosome 1 ("chr1" on the file) that is used to map the time points onto the nucleotide positions.

(C) Once formatted this way, data can then be displayed and processed using popular genomics tools such as the Integrative Genomic Viewer (IGV).

work, we show how mature standards can be recycled across research fields to save development time. To the best of our knowledge we document here the first case of a standard repurposing procedure between genomics and behavioral neuroscience.

The rationale for this work was our original observation that the storage of genomics data and the way these data are subsequently analyzed supports all the requirements of longitudinal data since both data types share a sequential structure. In a genomic sequence, nucleotides are ordered as a discrete series to which various layers of qualitative and quantitative annotation can be attached. This data structure, which is essentially a large table with a position in each line and various attributes in each column, can hold any sequentially organized information, including longitudinal recordings. By simply treating each nucleotide coordinate as a unit of time, we show that it is possible to use the most common genomic data formats for storing, visualizing, and analyzing behavioral information (including metadata) in a lossless fashion. What makes this data structure attractive for the handling of longitudinal information is not its specification but rather its ubiquitous use in genomics. In this field, the early adoption of strict data storage standards (Kent et al., 2002) has resulted in a huge number of tools built around formats agreed upon and supported by a community of thousands of laboratories around the world. Formats developed to define the position of genomic annotations therefore provide a perfect scaffold to store discrete time series of behavior. In a similar fashion, file formats implemented to represent scores along genomic sequences allow the storage of continuous time series of behavior (Figure 1). The available genomic tools (Afgan et al., 2018; Ernst and Kellis, 2012; Gentleman et al., 2004; Quinlan and Hall, 2010; Ramírez et al., 2016; Robinson and Thorvaldsdóttir, 2011; Zerbino et al., 2014) are also suitable to deal with time series because they allow both

sophisticated multi-scale visualization, genomic arithmetic operations required to conditionally extract and combine data portions, and complex post-processing techniques such as hidden Markov model (HMM) analysis.

RESULTS

We have developed Pergola (Python bEhavioRal GenOme tools LibrAry, http://cbcrg.github.io/pergola/) to demonstrate the feasibility of standard repurposing between genomics and behavioral neuroscience. Pergola is an open source tool capable of converting behavioral data into the most common genomic formats. This conversion merely requires a customizable mapping between the user's behavioral recordings and genomic data structures (e.g., time stamps mapped onto nucleotide positions, see Tables S1 and S2 and Supplemental Information for details). As a proof of principle, we have used Pergola to reprocess, visualize, and analyze worm, fly, and mouse data that had been obtained in previous studies using three different platforms and were stored in a variety of formats.

The first dataset corresponds to worm locomotion trajectories recorded with a video tracking system (Yemini et al., 2013). A part of the original study aimed to quantify the differences in locomotion between a mutant strain with reduced mobility and its wild-type control. From these data, we extracted motion states (discrete time intervals annotated as moving forward, backward, or paused) and per-frame body speed to BED and BedGraph formats, respectively, using Pergola, and visualized them on the Integrative Genomics Viewer (IGV) (Robinson and Thorvaldsdóttir, 2011). As this browser permits the simultaneous presentation of several annotation tracks, it lends itself well to the side-by-side display of worm recordings from different strains (Figure 2A). When doing so, it becomes straightforward to confirm that control N2 worms systematically reached higher speeds than the unc-16 individuals ("unc" stands for uncoordinated), an observation supported by the relative intensity of the speed signals displayed on the heatmap. This type of visualization is equally well suited for categorical (e.g., back or forward movement; Figure S1A) and continuous recordings (e.g., speed; Figure S1B). Using as input the individual speed trackings, we used deepTools (Ramírez et al., 2016), a popular suite for exploring sequencing data, to compute a principal component analysis. Our analysis not only discriminates mutants from their wild-type counterparts but also reflects the increased variability of wild-type worms with respect to the mutant group (Figure 2B), thus revealing individual variability as a potential confounding factor when performing group comparisons. Explorative analyses can be complemented with quantitative analyses using tools developed for genome arithmetic. Here, we reproduced the comparison of speed carried out in the original analysis (Figure 2C) by using BEDTools to extract intervals of the original speed trajectory fulfilling specific criteria (e.g., moving forward, Figure 3) and by aggregating the speed measurements within these intervals.

The second dataset is a collection of *Drosophila* behavioral trajectories processed using JAABA (Robie et al., 2017), a machine learning software for video tracking annotation. In the original article, the video recordings were processed with JAABA and further analyzed using custom MATLAB scripts. Here, we used Pergola to reformat the JAABA-annotated data into BED files. These files can be visualized using third-party genomics software like the Sushi R-package (Phanstiel et al., 2014) from Bioconductor (Figure 4A). Because Pergola is implemented as a Python library, it can be integrated with any of the Python tools developed for genomics analysis, including Pybedtools (Dale et al., 2011) (a wrapper for the BEDTools). To reproduce one of the original results of this study, we used Pybedtools to extract the regions annotated as chase behavior (a behavior that consists in closely following another individual for a certain amount of time) from the BED files and compared the chasing time across *Drosophila* groups with different genetic backgrounds (Figure 4B). Finally, to assess the relative contribution of the various locomotion features when annotating chase bouts, we adapted a volcano plot representation (Ritchie et al., 2015). In genomics, volcano plots are mostly used to display the difference of expression of a set of genes between two conditions. log2-fold changes are plotted against their log p values, thus allowing rapid identification of differences both meaningful and statistically significant. Such plots can be adapted to any experiment wherein a similar parameter is measured across different conditions, and whenever a sufficient level of replication allows statistical assessment. In the worm example, we considered every trajectory parameter (speed, orientation, angular velocity …) and measured the average log2-fold differences of these parameters between the intervals annotated as chase bouts and the remaining intervals. The p values were then obtained by considering all the intervals of a given type (chase or non-chase) across all individuals as replicates on which we ran a t test. The results (Figure 4C) identified angular speed to be one of the most discriminative features between chase and non-chase behavior.

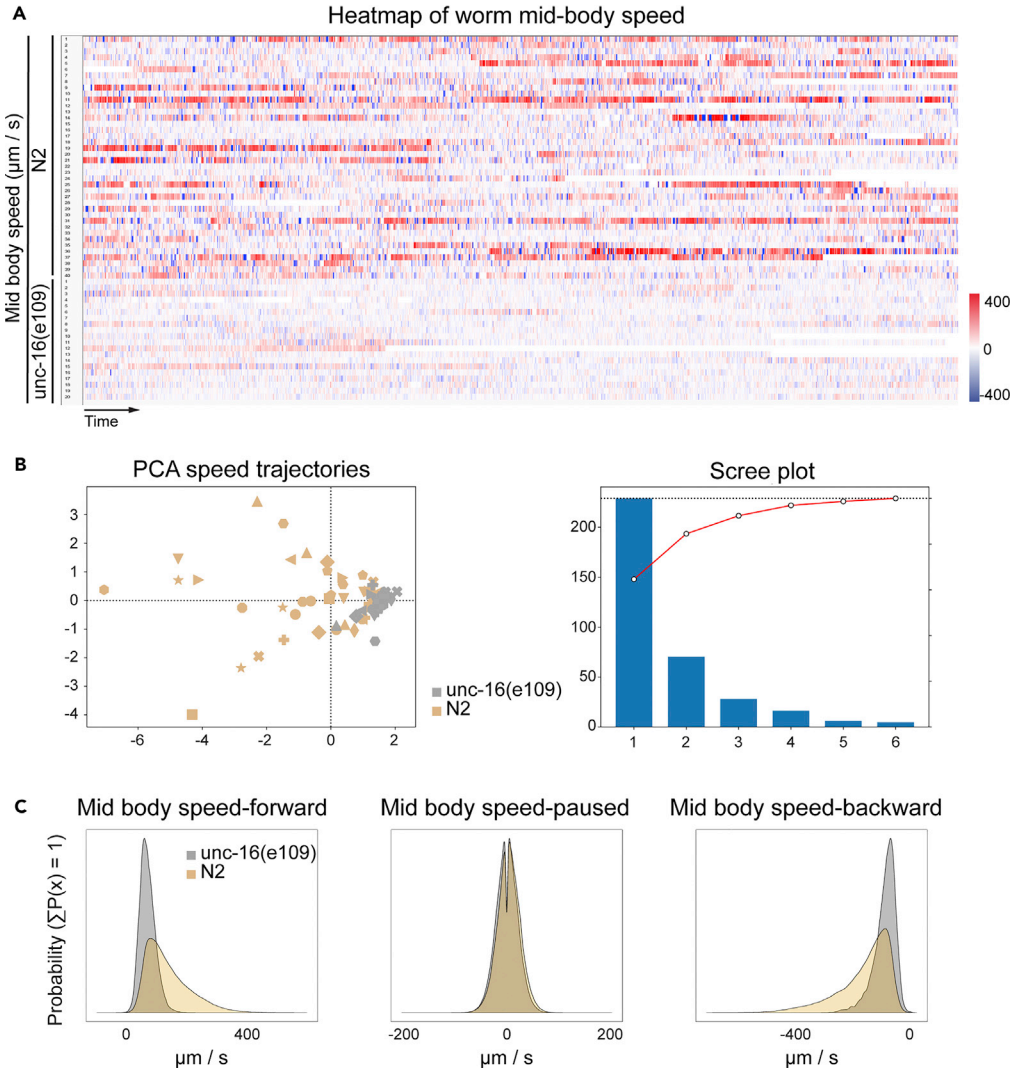


Figure 2. Repurposing of Genomic Software for the Analysis of *C. elegans* Locomotion Trajectories Using Pergola

(A) IGV snapshot of the comparison between *C. elegans* unc-16 mutants (20 tracks) and controls (40 tracks), where blue and red indicate the speed (in microns by frame) of the forward and backward movements, respectively. Tracks span 29,000 frames. Positive values (red) correspond to speeds measured during forward movements, whereas negative quantities indicate speed attained during backward movement.

(B) PCA and resulting screen plot (eigenvalues versus dimension) of binned speed trajectories obtained using deepTools. Geometric shapes depict single-worm trajectories (yellow for controls, gray for un-16 mutants). (Different geometric shapes are a built-in feature of the visualization and do not provide additional information.)

(C) Quantification of the speed distributions across the three possible motion states (forward, backward, and paused; same color code as in B).

See also Figures S1 and S7.

The third application of Pergola involves mice behavioral data that were obtained using PHECOMP cages (Espinosa-Carrasco et al., 2018). These cages enable fine-grained longitudinal monitoring of mice intake (solid and liquid), and can record a single individual during several weeks. In the original study, these devices were used to detect the influence of hypercaloric diets on meal patterns by comparing the feeding activity of a mice group exclusively offered a high-fat diet to their control group fed with standard chow. The sheer size of this dataset, with 9 weeks of recordings at a 1-s resolution, makes its exploration challenging.

The IGV browser used for the worm dataset does not allow interactive user-defined analysis (e.g., aggregation of data across time intervals). As an alternative, we therefore combined the Gviz Bioconductor

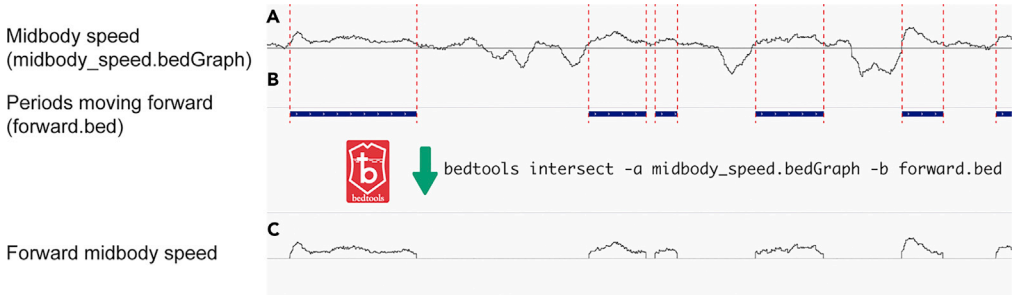


Figure 3. Use of BEDtools for Conditional Extraction of *C. elegans* Phenotypic Variables by Intersection with Motion State

(A and B) Tracks represent (A) the mid body speed of a single animal encoded in a BedGraph file (continuous annotation of behavior) and (B) the periods during the trajectory moving forward encoded in a BED file (discrete annotation of behavior).

(C) Phenotypic features in the BedGraph file can be intersected with direction states in the BED file using BEDTools. The resulting files contain only the regions of mid body speed overlapping with forward direction. Figure inspired by BEDTools documentation (http://bedtools.readthedocs.io/en/latest/content/tools/intersect.html).

R-package with the Shiny Visualization R-library (Figure S2) to assemble a suitable visualization tool. We used this interactive browser to explore the time-dependent behavioral changes in mice exposed to the high-fat diet (Figure 5) and complemented them with a standard heatmap analysis across the two conditions (Figure 6). To further investigate whether high-fat diet disrupted the circadian rhythm of mice, as previously shown (Kohsaka et al., 2007), we used deepTools to create a profile of the 24-hr feeding activity both group-wise (Figure 7) and individually (Figure S3), akin to an actogram. Although such analyses are totally standard in behavioral neurobiology, the aspect of this work most relevant to interoperability and reproducibility is that the actogram could be produced by a simple scripting of deepTools (cf. Transparent Methods) and that its packaging for further reuse without any need for extra coding is entirely supported by generic workflow managers such as Galaxy (Afgan et al., 2018).

When collecting longitudinal data, individual measurements are often non-independent of one another. The precise estimate of dependencies between successive events can therefore yield clues on the biological process underlying the generation of these observations. HMMs have long been known as methods of choice for analyzing such dependencies on behavioral time series (Arakawa et al., 2014; Carola et al., 2011; Yamato et al., 1992). HMMs were introduced in genomics in 1995 (Eddy, 1995) and since then, many off-the-shelf packages have been made available to carry this modeling. We selected chromHMM, a popular genomic software that infers chromatin states from chromatin marks (histone acetylation, CpG methylation, etc.) using an HMM (Ernst and Kellis, 2012). When doing so, individual chromatin marks are treated as probabilistic emissions of their underlying chromatin state. These marks constitute the input of the HMM training, whereas the output is a segmentation of the genome into regions having the same chromatin state. These states are the hidden variables. When training a model, the number of distinct hidden states has to be specified. HMM modeling is especially suitable when several states exist that can all emit the same observations but with different probabilities (just like loaded and fair dice can emit the same numbers but with different probabilities). The estimation of the transition probabilities across states as well as the individual emission probabilities of each state is named the training phase. It is usually an unsupervised process that explicitly relies on Bayes' theorem to estimate an HMM model whose joint probability with the data is maximized.

In the case of mice longitudinal feeding recordings, it is possible to treat every feeding bout as an emission, and one can then use HMM modeling to determine the typology of feeding behaviors most likely to have emitted the combinations of feeding bouts. Two parameters are needed to achieve this result: the number of hidden states (e.g., number of distinct feeding behaviors) and a description of the emissions produced by each state of the model (e.g., short feeding bout, long feeding bout). In this precise case, since the emissions are feeding bouts defined by their duration—a continuous value—it is common practice to do a discretization by binning the values into quantiles. In an ideal situation, both the hidden states and the discretized bouts should be mapped onto precise biological quantities or processes. In practice, however, even when this mapping is approximate, the resulting segmentation can reveal subtle changes in the structure of the data, thus allowing meaningful comparisons across datasets. As an illustration of this process,

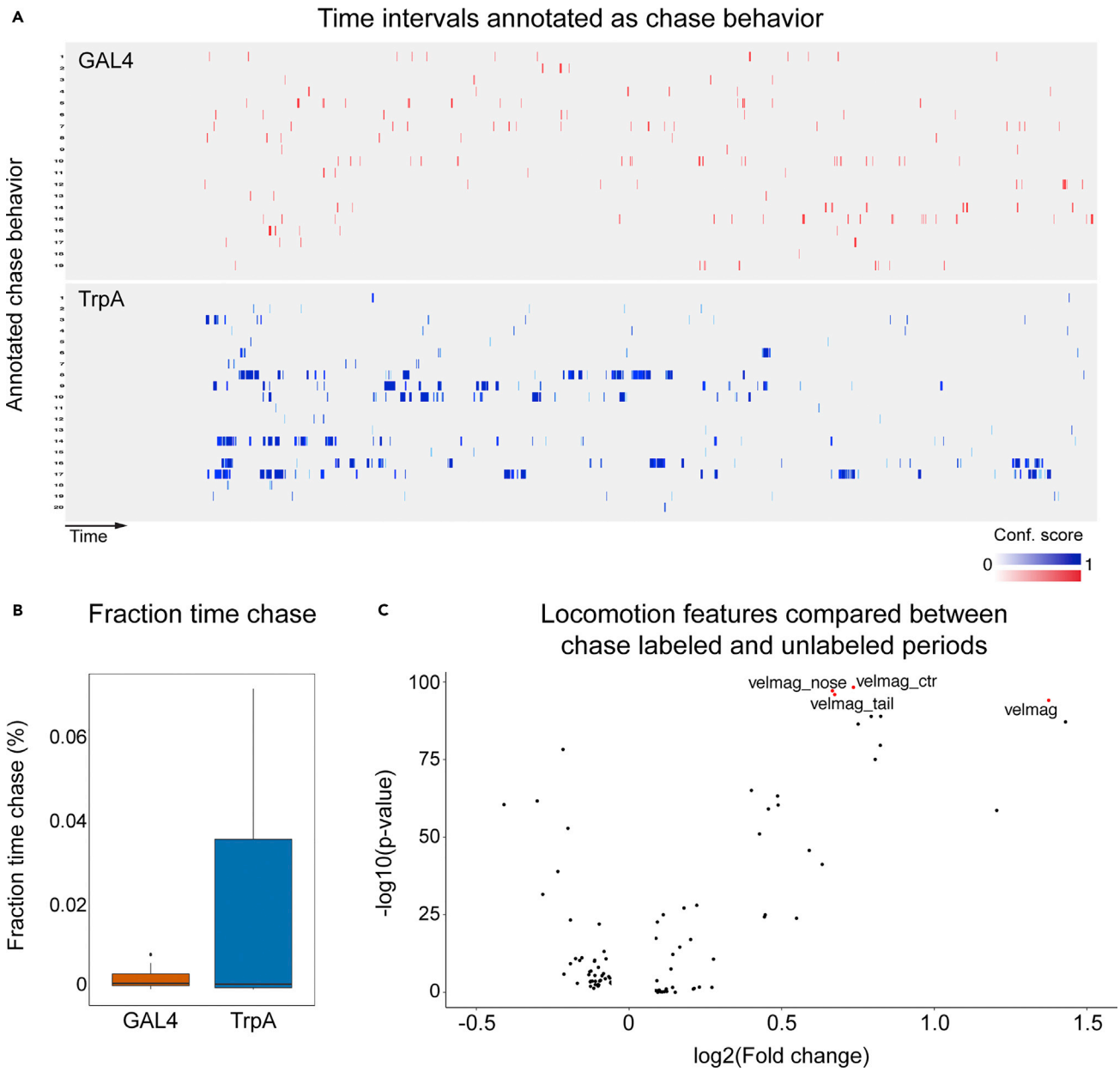


Figure 4. Analysis of *D. melanogaster* Locomotion Trajectories Using Pergola

(A) Fly chase behavior annotated by JAABA and rendered with Sushi. Chasing periods are represented as blue and red bars for the chase-prone and control flies, respectively. Color intensity indicates the confidence score for the behavioral classifier returned by JAABA.

(B) Boxplots represent the time fraction of the trajectories annotated as chase behavior (same color code as in B).

(C) In this volcano plot, each dot represents one of the parameters measured on a fly trajectory. The horizontal axis represents the average log fold change of this parameter between measurements made during periods annotated by JABBA as chase and non-chase. The vertical axis represents the p value (from t-test) of these differences. Parameters whose variation is both extreme and statistically significant (e.g., velmag: angular velocity) are colored in red. See also Figure S8.

we discretized the feeding bouts according to their duration in five quantiles, a number that roughly captures the various modes of bout distributions (Figure S4). We then used these quantiles as emissions and estimated a four-states model onto the data (Figure 8A). The result was a collection of intervals, each assigned to one of the four states. As expected, every bout duration was found to occur within every segmented interval, but with varying probabilities. Each state is characterized by a specific combination

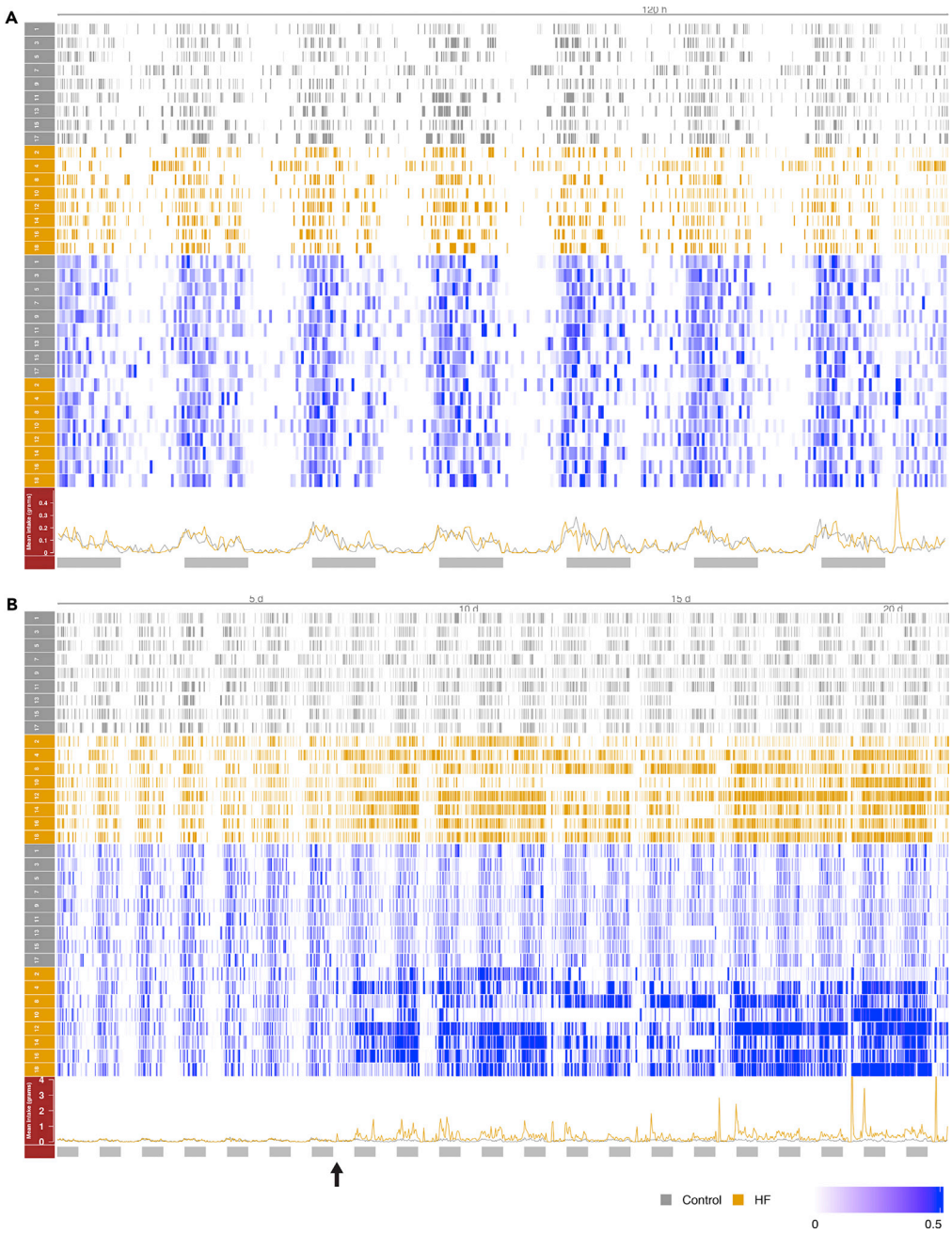


Figure 5. Shiny-Pergola Rendering of Mice Feeding Activity Showing Diet Influence on Meal Patterns

Feeding behavior during (A) the habituation phase (first week) and (B) the habituation phase and its transition, indicated by the arrow, to the first 2 weeks of the development of obesity in the high-fat mice (HF). The upper part of each panel (gray and orange) corresponds to raw meals depicted as rectangles proportional to their duration, whereas the middle part (blue tones) displays the accumulated food intakes of mice within 30-min time windows as a heatmap. In both cases, each row depicts data of a single individual. The bottom plot displays the group-wise average intake during these time windows. In all cases the gray is used for control mice and orange for HF mice. The zoomed snapshot (A) shows the micro-structure of feeding behavior during the habituation phase and how intakes are more frequent during dark periods in both mice groups (as mice are nocturnal animals). Substitution of standard chow by the high-fat food rapidly induces a disruption of the circadian feeding rhythmicity in the HF mice, as observed when browsing the period corresponding to the transition. In the heatmap, the darkest blue rectangles correspond to the maximum values: 0.5 g. See also Figures S2 and S9.

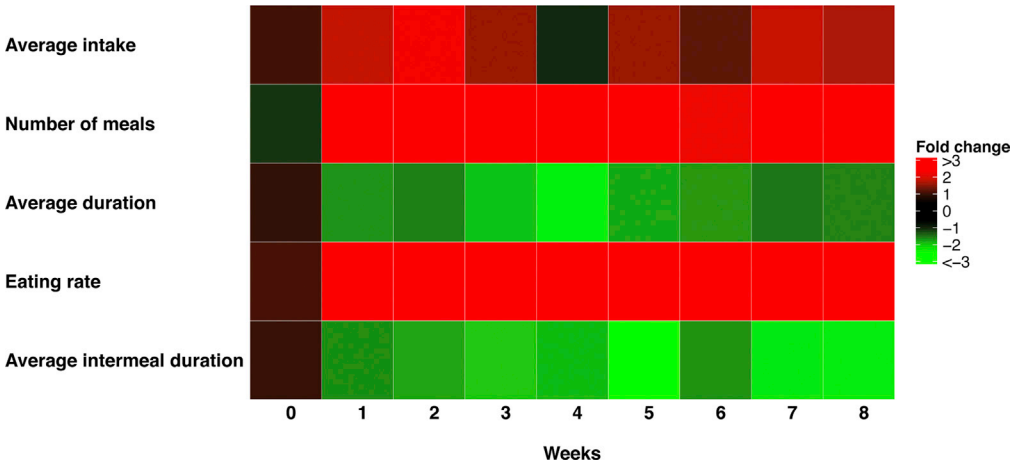


Figure 6. Heatmap Representation of the Fold Change in Feeding Behavior between Mice Exposed to High Fat and Their Control

Each square corresponds to the high-fat versus control mice fold change of the specific behavior (rows) during a given week (columns). Increased fold change >1 are depicted in red, whereas decreasing changes are shown in green. Color intensity is proportional to the fold change magnitude, and black indicates no variation. During the habituation phase (week 0), both groups of mice were fed with standard chow and no difference between their feeding behavior was observed. As soon as the high-fat diet is introduced, mice forced to eat exclusively this diet increased intake, number, and eating duration of meals and reduced the duration and separation between meals (animals in control group: 9, high-fat-diet group: 8).

of emission probabilities for the feeding bouts and transition probabilities across states (Figures S5 and S6, respectively).

When performing unsupervised modeling (i.e., starting from raw data without prior information), the states resulting from the training are merely statistical quantities whose combination maximizes the data probability. The biological meaning of these states remains to be explored. In the case at hand, most state labels turned out to be rather trivial to assign through visual inspection, with one of the states accounting for the absence of feeding activity, a second made of a combination of feeding bouts mostly coming from the short meals quantile, a third characterized by meals of a regular length, and finally, a fourth state enriched in long feeding bouts. Notably, although no information was provided to the model, the "short meals" state fits relatively well with binge eating behavior characteristic of compulsive-like feeding (Cottone et al., 2012). Binge eating is defined as the consumption of a large quantity of food in a very short period of time and seems to dominate the feeding activity of high-fat mice (green intervals in Figure 8A). Taking advantage of the BED formatting, we used Pybedtools to compare the distribution of states across the circadian cycle in basal conditions (habituation phase) and after the introduction of the high-fat food (obesity development phase, Figure 8B) with the resulting representation supporting the notion that high-fat-fed mice increase periods of short meals at the expense of inactive periods. This analysis also reveals the tendency of these changes to occur during the light phases of the day when mice should be less active owing to their nocturnal nature. Last but not least, the HMM decoding makes it possible to highlight probable artifacts such as the unusual high feeding activity observed in some control mice (Figure 8A, blue patch in the middle of the trajectory of control mice #4 and #9).

The three examples of analysis of different model organisms based on distinct data processing procedures clearly illustrate the versatility brought about by genomics standards to behavioral analysis. Yet, these standards not only cater to a wide range of different analyses but also allow uniform operations to be carried out across datasets. For instance, we provided a proof of principle of how the three visualization procedures can be indistinctly applied onto the three datasets, regardless of the organism or the recording platform (Figures S7–S9). Altogether these analyses demonstrate how the use of genomics standards allows an unprecedented amount of interoperability across methods and datasets when dealing with behavioral time series data.

To ensure computational reproducibility, we ran Pergola using Nextflow (Di Tommaso et al., 2017), a workflow manager that allows entire pipelines to be deployed using software containers. The main advantage of

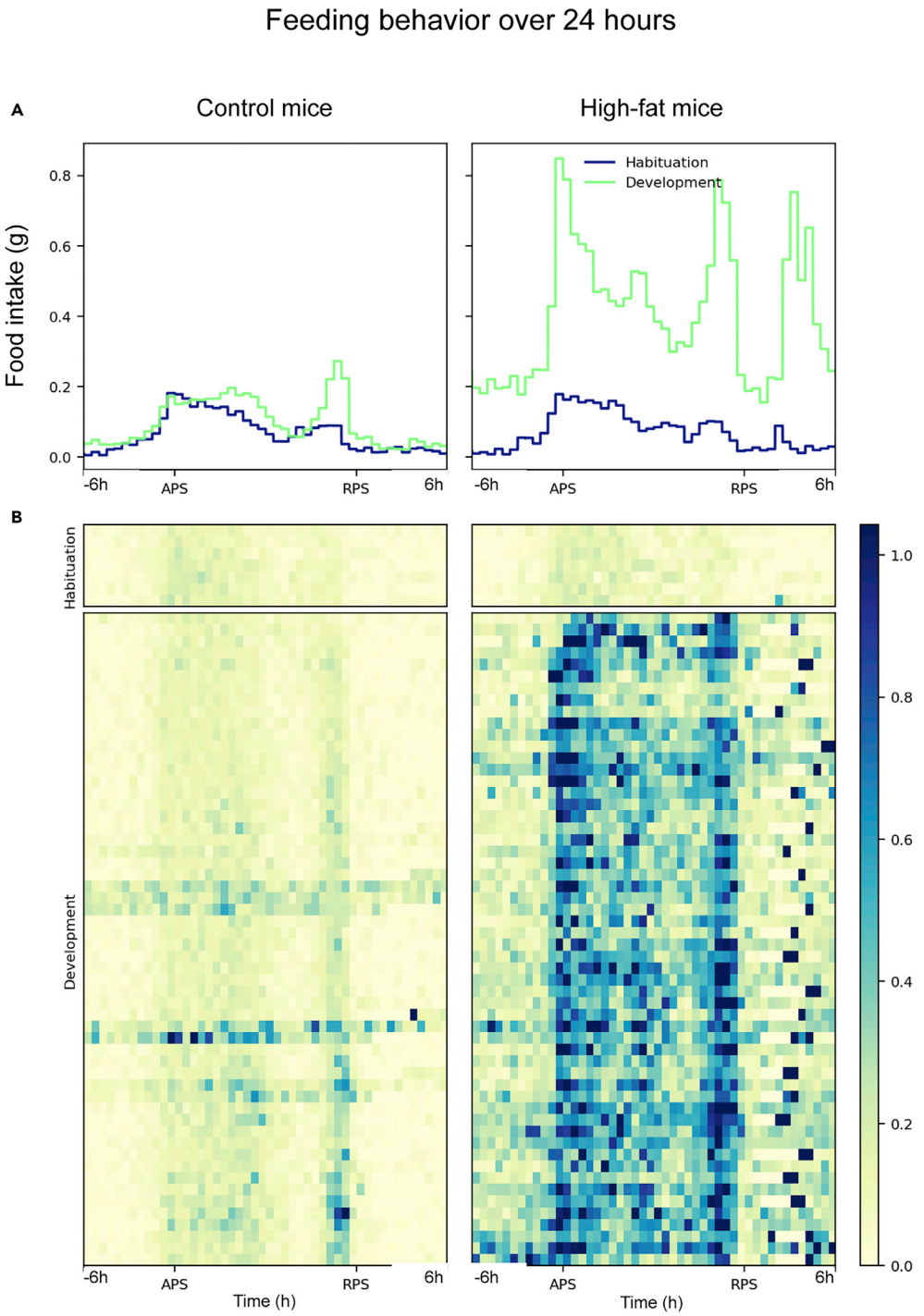


Figure 7. Mouse Circadian Profiles of Feeding Activity Obtained Using deepTools

(A) An actogram summarizing group-wise feeding activities generated using deepTools. Left panel corresponds to control mice, and the right panel corresponds to high-fat-diet mice. The x axis represents the 24-hr daily cycle of behavior across the light/dark phases with the active phase start (APS) and the resting phase start (RPS) representing points wherein lights were switched off and on, respectively. The y axis represents the group average food intake, averaged over all the days of the habituation phase (blue line) or development phase (green line).

(B) Day-by-day recordings (averaged over all mice in a group) with every line representing a day, ordered by date with the early ones on the top. Food intake was normalized across all recordings, with dark blue indicating the highest values. See also Figure S3.

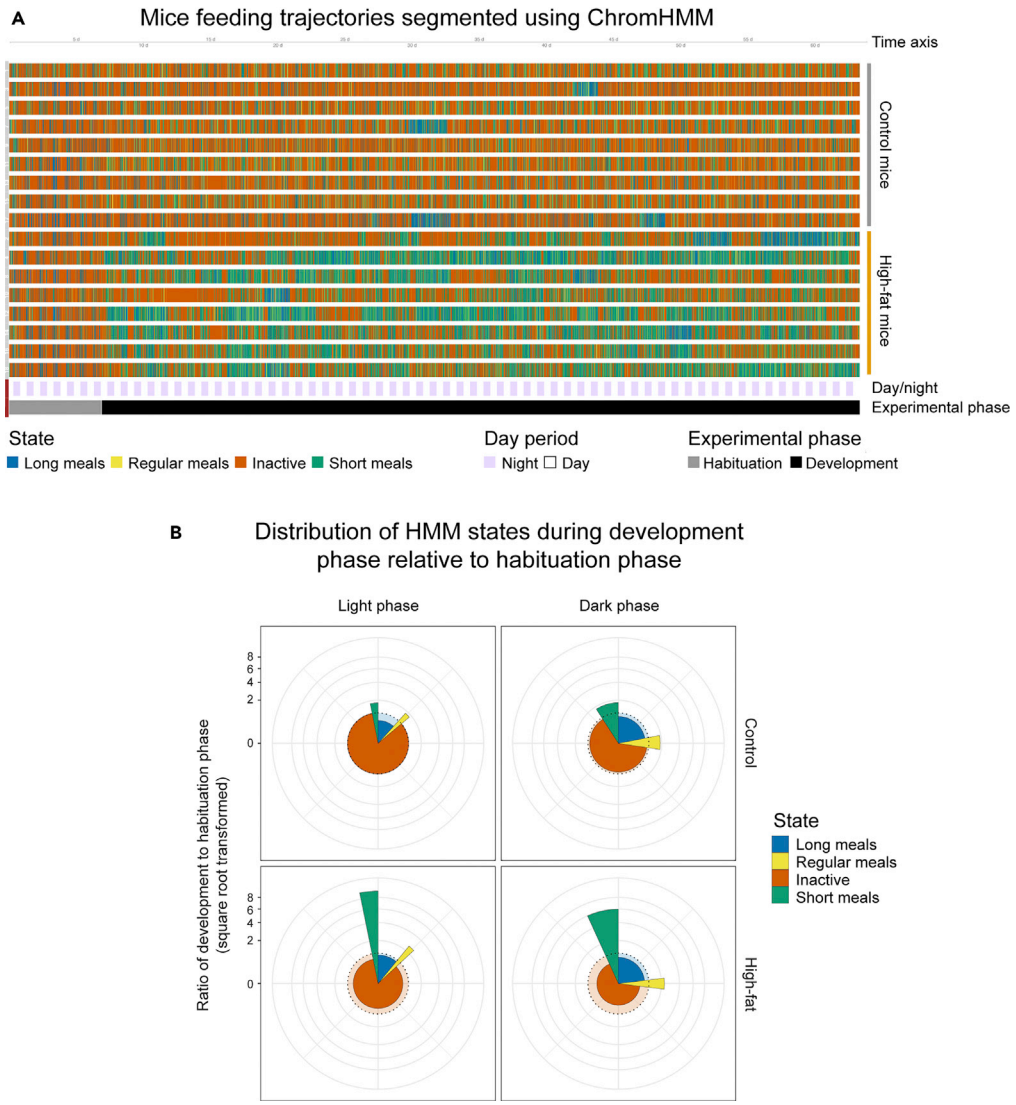


Figure 8. Profiles of Mice Feeding Behavior Segmented Using ChromHMM

(A) Mice feeding bouts were distributed in five quintiles (see Figure S4) and eventually used to train a four-state HMM model thus resulting in each trajectory segment (300 s) to be probabilistically assigned to one of the four states. The most common feeding bout duration observed in each state was used to label it (long meals, regular meals, short meals, and inactive). A day/night track indicates dark (violet) and light phases (white) of the circadian rhythm. The last track is colored to show habituation (gray) and obesity development (black) phases of the experiment.

(B) In the spie chart representation, every slice represents the fraction of time spent in any of the four HMM states during light and day phases (left and right) and for the control and the high-fat animals (top and bottom). On these charts, the angular spread of each slice represents the relative duration of the considered state during habituation (i.e., standard chow), whereas the radial extent of the same slice represents the ratio between a measurement made during development and habituation. For instance, the green slice representing high-fat mice daily intake (bottom left) is the one showing the strongest difference between the habituation and the development phases but represents only a small fraction of the state distribution during habituation (see also Figures S4–S6).

containers is that they bundle together software that can run in the same fashion irrespective of the computational environment. Containers can also be handled by workflow managers (Di Tommaso et al., 2015) like Nextflow that provide increased scalability through optimized parallelization. In order for analyses to become easily shareable and reproducible, however, containers, software, and data must all be made available through public repositories (e.g., Docker-Hub, GitHub, and Zenodo, respectively). Thanks to the integration of these resources within Nextflow, all the analyses shown here can be repeated by entering a single

command line (Transparent Methods) on an adequately configured computer (i.e., Nextflow and Docker must be installed so as to allow software to be automatically gathered and deployed). Community-wide adoption of such a global computational strategy could be a major step toward the implementation of the FAIR principle (Wilkinson et al., 2016) in longitudinal behavioral studies.

DISCUSSION

Pergola is a simple yet effective solution to the problem of data interoperability in behavioral neuroscience. By providing access to an established data standard, Pergola's data harmonization scheme has the potential to increase interoperability and computational reproducibility. The need for these improvements has become critical now that neuroscience is entering a data-intensive cycle (Wiener et al., 2016). Existing commercial and open source behavioral software often provide a toolkit designed to perform a specific set of analyses, but these toolkits are often hard to adapt and frequently bound to a specific acquisition platform. By contrast, Pergola enables a flexible framework in which ad hoc computational solutions can be rapidly implemented by re-adapting existing tools designed to deal with genomics standards. The benefits are non-neglectable since genomics offers a rapidly growing corpus of statistical analysis software (Gentleman et al., 2004; Huber et al., 2015). In this work, we show on three distinct model systems how genomic tools can be easily applied for reproducing common behavioral analysis. The simplest, albeit arguably the most powerful aspect of the reliance on genomics standards is visualization. Visualization tools are complex software whose user interface is essential for effective data exploration. Multiscale support is especially important when zooming in and out of large datasets exhibiting meaningful signals at various levels of granularity. We show here that genomics not only provides a ready-made solution but also allows for visualization procedures that are shareable and re-usable across a wide range of different analyses. Our Shiny-based browser is a striking example of how effective and problem-tailored shareable visualization procedures can be implemented in behavioral biology, following the trends set in genomics (Cao et al., 2018).

Genomics can provide a substantial number of advanced analysis techniques readily implemented (Gentleman et al., 2004; Quinlan and Hall, 2010; Ernst and Kellis, 2012; Ramírez et al., 2016) for the study of animal behavior. Thanks to its critical mass and the development of very large flagship projects like ENCODE or GTEx (The ENCODE Project Consortium et al., 2012; Ardlie et al., 2015), genomics software tools tend to be more mature and better supported than their counterparts in other research fields. In this work, we show how we can integrate chromHMM (Ernst and Kellis, 2012), a tool that has been extensively used by ENCODE to characterize chromatin regions (Ernst and Kellis, 2010; The ENCODE Project Consortium et al., 2012; Yue et al., 2014), to annotate mice feeding activity. Likewise, we adopted deepTools (Ramírez et al., 2016) functionalities to cluster posture-tracking-derived data from *C. elegans* and to derive mouse circadian profiles. The latter constitutes a compelling example of how to implement a useful data representation with little computational proficiency. In addition, we showed how an analysis strategy commonly used for genome-wide gene expression data, the volcano plot (Ritchie et al., 2015; Liu et al., 2018), can be used to identify the main behavioral features, leading to the differential annotation of a given behavior.

In par with visualization, the other main benefit of the standard repurposing presented here is probably increased interoperability. Interoperability stems from datasets being standardized into the same format and therefore operable with the same tools. As a consequence, it becomes easier to share, compare, and reuse data (Sansone et al., 2012) and apply common analysis procedures to heterogeneous datasets. Increased interoperability allows laboratories to work under open-data collaborative scientific environments with major benefits for the community (Gomez-Marin et al., 2014). Last but not least, interoperability offers unified and flexible visualizations of diverse recordings that may include non-behavioral data. For instance, in the near future, the simultaneous rendering of behavioral trajectories along with environmental annotations or electrophysiological recordings may allow to identify essential explanatory patterns on the data (Anderson and Perona, 2014).

The technical feasibility of interoperability comes, however, with extra caveat. More specifically, it must be stressed that meaningful comparison of alternative recordings depends on a good understanding of their relative timescales (Brown and de Bivort, 2018). Although our approach makes it possible to process through the same channels behavioral features lasting just milliseconds (e.g., response to an odor stimulus in the fly [Gaudry et al., 2013]) or a day (e.g., circadian rhythms [Geissmann et al., 2017]), the decision as to whether such analyses can be meaningfully compared remains a matter for scientific debate and analysis.

Interoperability, nonetheless, offers a unique window of opportunity to explore relationship between short- and long-term behavioral patterns. Inter-operating data across various species also brings new possibilities, such as the evolutionary treatment of behavioral characters and their interpretation in terms of homology (Gomez-Marin et al., 2016). Of course, in contrast with nucleotide homology where genomic positions are inferred to be homologous to one another, behavioral homology is more similar to character-based phylogeny and may have to be inferred by comparing secondary models of longitudinal data (e.g., HMMs) rather than the longitudinal data itself. Similar analyses have been carried out in the field of social sciences, with complex longitudinal studies recorded as simple sequences and processed using mathematical tools inspired from evolutionary biology (Gauthier et al., 2010, 2009).

We also demonstrate here how high-throughput behavioral analysis can become more systematic and reproducible by adopting workflow managers such as Nextflow (Di Tommaso et al., 2017). The task of deploying analysis relying on such managers can be highly simplified by adopting mature genomic solutions. Additional benefit comes from the large user and developer communities and the rigorous version control of these tools. The robust handling of changing versions of the same software will be achieved by packaging them into software containers. Besides, these tools share input and output file formats, which allows to develop orchestrated pipelines.

This cross-fertilization between scientific fields provides a prime example of how interdisciplinarity can save development time by allowing collections of extensively validated software and methods to be shared and mutualized. Indeed, the problem of massive data integration is not specific to neuroscience, with novel types of longitudinal recordings being collected in hospitals (Jensen et al., 2014) and societal contexts (de Montjoye et al., 2015). Although properly analyzing these data will pose a challenge, its expected impact on human health will definitely make it worthwhile (Jensen et al., 2014; Price et al., 2017).

Limitations of the Study

Pergola is not meant to integrate genomics and behavioral data. In this study, genomic format capacities are merely used as information technology solutions for the programmatic and standardized management of behavioral data and subsequent statistical analysis. The nucleotide alphabet does not contribute to the encoding process that ignores the biological nature of DNA. As a consequence, the encoded behavioral data cannot be used to infer functional and evolutionary relationships across species and experiments through standard evolutionary-based genome alignment tools.

METHODS

All methods can be found in the accompanying Transparent Methods supplemental file.

SUPPLEMENTAL INFORMATION

Supplemental Information includes Transparent Methods, nine figures, and two tables and can be found with this article online at https://doi.org/10.1016/j.isci.2018.10.023.

ACKNOWLEDGMENTS

We wish to thank P. Di Tommaso for support in the implementation of Nextflow pipelines, M. Fructuoso for helpful suggestions and comments, and T. Ferrar for manuscript revision and helpful comments. We acknowledge the support of the "Secretaria d'Universitats i Recerca del Departament d'Economia i Coneixement de la Generalitat i del Fons Social Europeu," Spanish Ministry of Economy, Industry and Competitiveness (MEIC) under grant numbers BFU2014-55062-P and BFU2017-88264-P, the EMBL partnership, FEDER, the "Centro de Excelencia Severo Ochoa" Programme, and the CERCA Programme/Generalitat de Catalunya. This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 635290-PanCanRisk and OpenRiskNet_731076. This reflects only the author's view and the funder is not responsible for any use that may be made of the information it contains. M.D. is supported by DIUE de la Generalitat de Catalunya (Grups consolidats SGR 2017/926), the Fondation Jérôme Lejeune (Paris, France), MINECO (SAF2013-49129-C2-1-R; SAF2016-79956-R), CDTI ("Smartfoods") and EU (EraNet Neuron PCIN-2013-060 and JPND AC17/00006), and the Catalan foundation "La Marató de TV3" (#2016/20-30). The CIBER of Rare Diseases is an initiative of the ISCIII.

AUTHOR CONTRIBUTIONS

C.N. and M.D. directed the work; C.N., M.D., I.E., and J.E.-C. contributed ideas and wrote the manuscript; C.N. and J.E.-C. designed the software; J.E.-C. wrote the software and analyzed data; and J.P., T.H.P., and J.E.-C. were involved in web server implementation and Galaxy analysis.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: April 25, 2018
Revised: September 12, 2018
Accepted: October 22, 2018
Published: November 30, 2018

REFERENCES

Afgan, E., Baker, D., Batut, B., van den Beek, M., Bouvier, D., Cech, M., Chilton, J., Clements, D., Coraor, N., Grüning, B.A., et al. (2018). The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. Nucleic Acids Res. *46*, W537–W544.

Anderson, D.J., and Perona, P. (2014). Toward a science of computational ethology. Neuron *84*, 18–31.

Arakawa, T., Tanave, A., Ikeuchi, S., Takahashi, A., Kakihara, S., Kimura, S., Sugimoto, H., Asada, N., Shiroishi, T., Tomihara, K., et al. (2014). A male-specific QTL for social interaction behavior in mice mapped with automated pattern detection by a hidden Markov model incorporated into newly developed freeware. J. Neurosci. Methods *234*, 127–134.

Ardlie, K.G., Deluca, D.S., Segre, A.V., Sullivan, T.J., Young, T.R., Gelfand, E.T., Trowbridge, C.A., Maller, J.B., Tukiainen, T., Lek, M., et al. (2015). The Genotype-Tissue Expression (GTEx) pilot analysis: multitissue gene regulation in humans. Science *348*, 648–660.

Brown, A.E.X., and de Bivort, B. (2018). Ethology as a physical science. Nat. Phys. https://doi.org/10.1038/s41567-018-0093-0.

Cao, X., Yan, Z., Wu, Q., Zheng, A., and Zhong, S. (2018). GIVE: portable genome browsers for personal websites. Genome Biol. *19*, 92.

Carola, V., Mirabeau, O., and Gross, C.T. (2011). Hidden Markov model analysis of maternal behavior patterns in inbred and reciprocal hybrid mice. PLoS One *6*, e14753.

Cottone, P., Wang, X., Park, J.W., Valenza, M., Blasio, A., Kwak, J., Iyer, M.R., Steardo, L., Rice, K.C., Hayashi, T., and Sabino, V. (2012). Antagonism of sigma-1 receptors blocks compulsive-like eating. Neuropsychopharmacology *37*, 2593–2604.

Dale, R.K., Pedersen, B.S., and Quinlan, A.R. (2011). Pybedtools: a flexible Python library for manipulating genomic datasets and annotations. Bioinformatics *27*, 3423–3424.

de Montjoye, Y.-A., Radaelli, L., Singh, V.K., and Pentland, A.S. (2015). Identity and privacy. Unique in the shopping mall: on the reidentifiability of credit card metadata. Science *347*, 536–539.

Di Tommaso, P., Chatzou, M., Floden, E.W., Barja, P.P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. Nat. Biotechnol. *35*, 316–319.

Di Tommaso, P., Palumbo, E., Chatzou, M., Prieto, P., Heuer, M.L., and Notredame, C. (2015). The impact of Docker containers on the performance of genomic pipelines. PeerJ *3*, e1273.

Eddy, S.R. (1995). Multiple alignment using hidden Markov models. Proc. Int. Conf. Intell. Syst. Mol. Biol. *3*, 114–120.

Ernst, J., and Kellis, M. (2012). ChromHMM: automating chromatin-state discovery and characterization. Nat. Methods *9*, 215–216.

Ernst, J., and Kellis, M. (2010). Discovery and characterization of chromatin states for systematic annotation of the human genome. Nat. Biotechnol. *28*, 817–825.

Espinosa-Carrasco, J., Burokas, A., Fructuoso, M., Erb, I., Martín-García, E., Gutiérrez-Martos, M., Notredame, C., Maldonado, R., and Dierssen, M. (2018). Time-course and dynamics of obesity-related behavioral changes induced by energy-dense foods in mice. Addict. Biol. *23*, 531–543.

Field, D., Amaral-Zettler, L., Cochrane, G., Cole, J.R., Dawyndt, P., Garrity, G.M., Gilbert, J., Glöckner, F.O., Hirschman, L., Karsch-Mizrachi, I., et al. (2011). The genomic standards consortium. PLoS Biol. *9*, e1001088.

Gaudry, Q., Hong, E.J., Kain, J., de Bivort, B.L., and Wilson, R.I. (2013). Asymmetric neurotransmitter release enables rapid odour lateralization in Drosophila. Nature *493*, 424–428.

Gauthier, J.-A., Widmer, E.D., Bucher, P., and Notredame, C. (2010). Multichannel sequence analysis applied to social science data. Sociol. Methodol. *40*, 1–38.

Gauthier, J.-A., Widmer, E.D., Bucher, P., and Notredame, C. (2009). How much does it cost?: Optimization of costs in sequence analysis of social science data. Sociol. Methods Res. *38*, 197–231.

Geissmann, Q., Garcia Rodriguez, L., Beckwith, E.J., French, A.S., Jamasb, A.R., and Gilestro, G.F. (2017). Ethoscopes: an open platform for high-throughput ethomics. PLoS Biol. *15*, e2003026.

Gentleman, R.C., Carey, V.J., Bates, D.M., Bolstad, B., Dettling, M., Dudoit, S., Ellis, B., Gautier, L., Ge, Y., Gentry, J., et al. (2004). Bioconductor: open software development for computational biology and bioinformatics. Genome Biol. *5*, R80.

Gomez-Marin, A., Oron, E., Gakamsky, A., Dan Valente, Benjamini, Y., and Golani, I. (2016). Generative rules of Drosophila locomotor behavior as a candidate homology across phyla. Sci. Rep. *6*, 27555.

Gomez-Marin, A., Paton, J.J., Kampff, A.R., Costa, R.M., and Mainen, Z.F. (2014). Big behavioral data: psychology, ethology and the foundations of neuroscience. Nat. Neurosci. *17*, 1455–1462.

Hong, W., Kennedy, A., Burgos-Artizzu, X.P., Zelikowsky, M., Navonne, S.G., Perona, P., and Anderson, D.J. (2015). Automated measurement of mouse social behaviors using depth sensing, video tracking, and machine learning. Proc. Natl. Acad. Sci. U S A *112*, E5351–E5360.

Huber, W., Carey, V.J., Gentleman, R., Anders, S., Carlson, M., Carvalho, B.S., Bravo, H.C., Davis, S., Gatto, L., Girke, T., et al. (2015). Orchestrating high-throughput genomic analysis with Bioconductor. Nat. Methods *12*, 115–121.

Jensen, A.B., Moseley, P.L., Oprea, T.I., Ellesøe, S.G., Eriksson, R., Schmock, H., Jensen, P.B., Jensen, L.J., and Brunak, S. (2014). Temporal disease trajectories condensed from population-wide registry data covering 6.2 million patients. Nat. Commun. *5*, 1–10.

Karsch-Mizrachi, I., Takagi, T., and Cochrane, G.; International Nucleotide Sequence Database Collaboration (2018). The international nucleotide sequence database collaboration. Nucleic Acids Res. *46*, D48–D51.

Kent, W.J., Sugnet, C.W., Furey, T.S., Roskin, K.M., Pringle, T.H., Zahler, A.M., and Haussler, D. (2002). The human genome browser at UCSC. Genome Res. *12*, 996–1006.

Kohsaka, A., Laposky, A.D., Ramsey, K.M., Estrada, C., Joshu, C., Kobayashi, Y., Turek, F.W., and Bass, J. (2007). High-fat diet disrupts

behavioral and molecular circadian rhythms in mice. Cell Metab. *6*, 414–421.

Liu, N., Lee, C.H., Swigut, T., Grow, E., Gu, B., Bassik, M.C., and Wysocka, J. (2018). Selective silencing of euchromatic L1s revealed by genome-wide screens for L1 regulators. Nature *553*, 228–232.

Munafò, M.R., Nosek, B.A., Bishop, D.V.M., Button, K.S., Chambers, C.D., du Sert, N.P., Simonsohn, U., Wagenmakers, E.-J., Ware, J.J., and Ioannidis, J.P.A. (2017). A manifesto for reproducible science. Nat. Hum. Behav. *1*, 0021.

Pérez-Escudero, A., Vicente-Page, J., Hinz, R.C., Arganda, S., and de Polavieja, G.G. (2014). idTracker: tracking individuals in a group by automatic identification of unmarked animals. Nat. Methods *11*, 743–748.

Phanstiel, D.H., Boyle, A.P., Araya, C.L., and Snyder, M.P. (2014). Sushi.R: flexible, quantitative and integrative genomic visualizations for publication-quality multi-panel figures. Bioinformatics *30*, 2808–2810.

Price, N.D., Magis, A.T., Earls, J.C., Glusman, G., Levy, R., Lausted, C., McDonald, D.T., Kusebauch, U., Moss, C.L., Zhou, Y., et al. (2017). A wellness study of 108 individuals using personal, dense, dynamic data clouds. Nat. Biotechnol. *35*, 747–756.

Quinlan, A.R., and Hall, I.M. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics *26*, 841–842.

Ramírez, F., Ryan, D.P., Grüning, B., Bhardwaj, V., Kilpert, F., Richter, A.S., Heyne, S., Dündar, F., and Manke, T. (2016). deepTools2: a next generation web server for deep-sequencing data analysis. Nucleic Acids Res. *44*, W160–W165.

Ritchie, M.E., Phipson, B., Wu, D., Hu, Y., Law, C.W., Shi, W., and Smyth, G.K. (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res. *43*, e47.

Robie, A.A., Hirokawa, J., Edwards, A.W., Umayam, L.A., Lee, A., Phillips, M.L., Card, G.M., Korff, W., Rubin, G.M., Simpson, J.H., et al. (2017). Mapping the neural substrates of behavior. Cell *170*, 393–406.e28.

Robinson, J.T., and Thorvaldsdóttir, H. (2011). Integrative genomics viewer. Nature *29*, 24–26.

Sansone, S.-A., Rocca-Serra, P., Field, D., Maguire, E., Taylor, C., Hofmann, O., Fang, H., Neumann, S., Tong, W., Amaral-Zettler, L., et al. (2012). Toward interoperable bioscience data. Nat. Genet. *44*, 121–126.

Schaefer, A.T., and Claridge-Chang, A. (2012). The surveillance state of behavioral automation. Curr. Opin. Neurobiol. *22*, 170–176.

Solt, L.A., Wang, Y., Banerjee, S., Hughes, T., Kojetin, D.J., Lundasen, T., Shin, Y., Liu, J., Cameron, M.D., Noel, R., et al. (2012). Regulation of circadian behaviour and metabolism by synthetic REV-ERB agonists. Nature *485*, 62–68.

Suez, J., Korem, T., Zeevi, D., Zilberman-Schapira, G., Thaiss, C.A., Maza, O., Israeli, D., Zmora, N., Gilad, S., Weinberger, A., et al. (2014). Artificial sweeteners induce glucose intolerance by altering the gut microbiota. Nature *514*, 181–186.

The ENCODE Project Consortium, Dunham, I., Kundaje, A., Aldred, S.F., Collins, P.J., Davis, C.A., Doyle, F., Epstein, C.B., Frietze, S., Harrow, J., Kaul, R., et al. (2012). An integrated encyclopedia of DNA elements in the human genome. Nature *489*, 57–74.

Wiener, M., Sommer, F.T., Ives, Z.G., Poldrack, R.A., and Litt, B. (2016). Enabling an open data ecosystem for the neurosciences. Neuron *92*, 929.

Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L.B., Bourne, P.E., et al. (2016). The FAIR guiding principles for scientific data management and stewardship. Sci. Data *3*, 160018.

Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden Markov model, in: Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 379–385.

Yemini, E., Jucikas, T., Grundy, L.J., Brown, A.E.X., and Schafer, W.R. (2013). A database of *Caenorhabditis elegans* behavioral phenotypes. Nat. Methods *10*, 877–879.

Yue, F., Cheng, Y., Breschi, A., Vierstra, J., Wu, W., Ryba, T., Sandstrom, R., Ma, Z., Davis, C., Pope, B.D., et al.; Mouse ENCODE Consortium (2014). A comparative encyclopedia of DNA elements in the mouse genome. Nature *515*, 355–364.

Zerbino, D.R., Johnson, N., Juettemann, T., Wilder, S.P., and Flicek, P. (2014). WiggleTools: parallel processing of large collections of genome-wide datasets for visualization and statistical analysis. Bioinformatics *30*, 1008–1009.

ISCI, Volume *9*

Supplemental Information

Pergola: Boosting Visualization and Analysis of Longitudinal Data by Unlocking Genomic Analysis Tools

Jose Espinosa-Carrasco, Ionas Erb, Toni Hermoso Pulido, Julia Ponomarenko, Mara Dierssen, and Cedric Notredame

Supplemental Information

Supplemental Figures

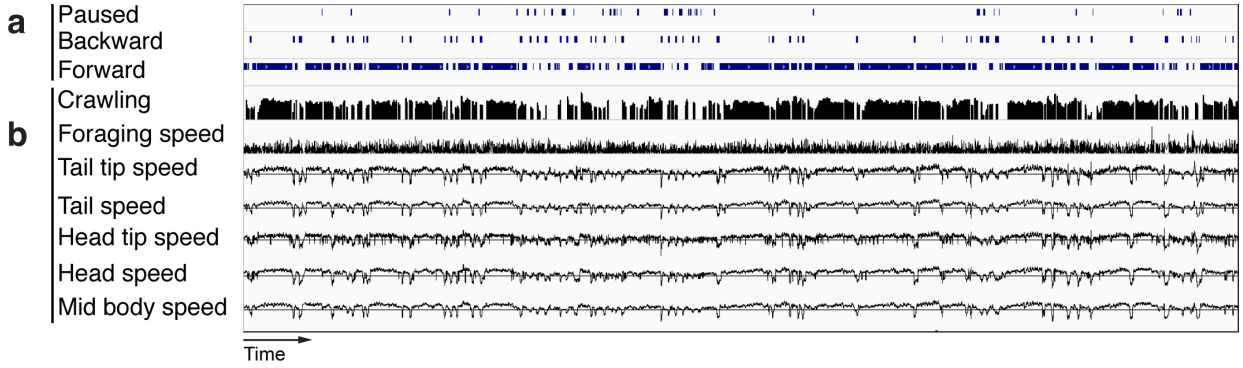


Figure S1 - *C. elegans* motor behavior visualization using Pergola and the IGV genomic browser. **(a)** Each blue track represents the intervals corresponding to each motion state (paused, backward or forward) of a single N2 individual. **(b)** Measures of several locomotion types on the same individual: crawling amplitude (0, 50), foraging speed (0, 730), tail tip, tail, head, and mid body speed (-700, 700) and head tip speed (-1050, 810). Numbers in parenthesis indicate the displayed variable range. All speed measures are expressed in microns/second except crawling amplitude expressed in degrees/second and foraging speed in degrees. Related to Figure 1.

Behavioral browser

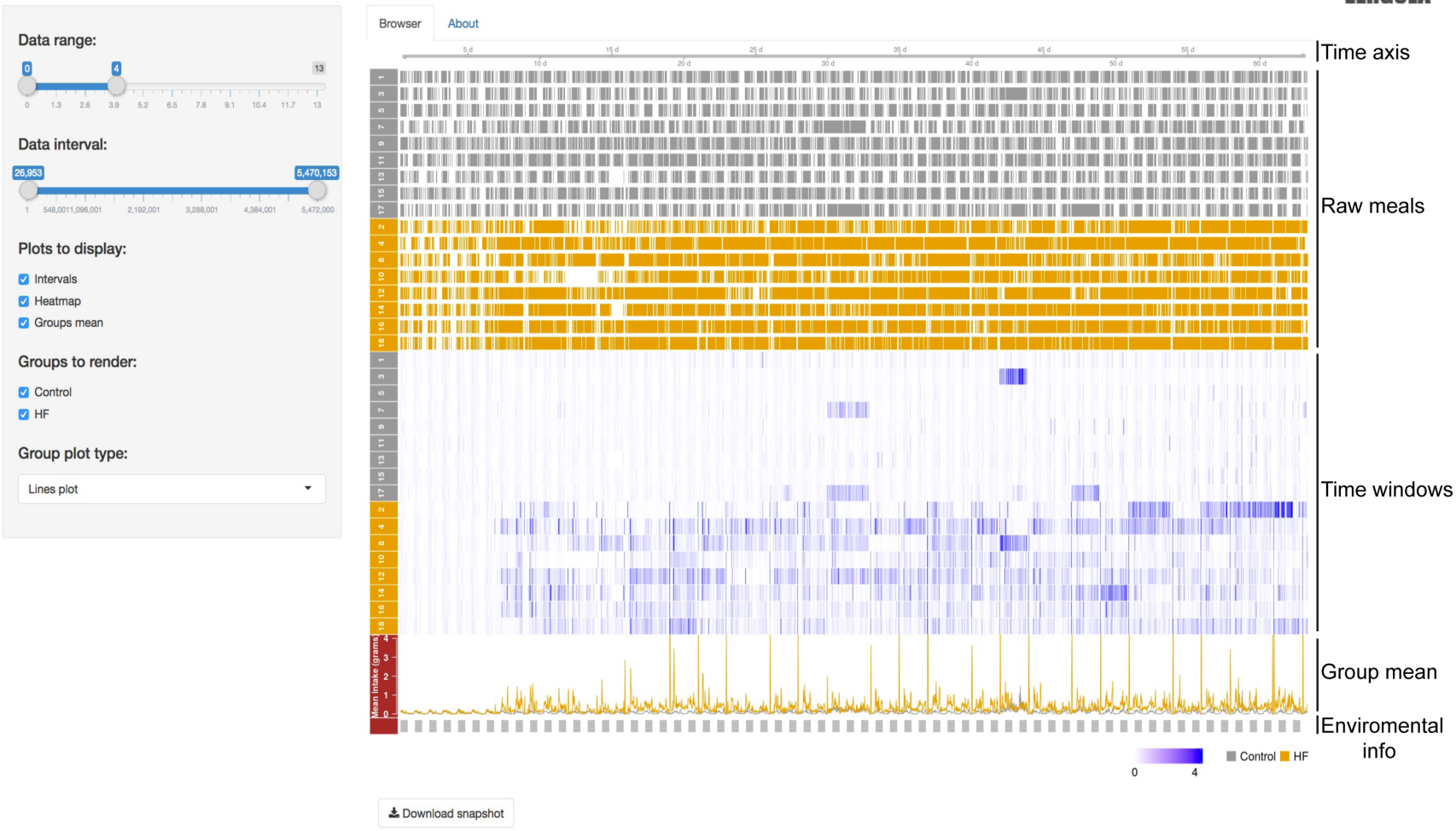


Figure S2 - Screenshot of visualization of a 9 weeks mice experiment using Shiny-Pergola. The Shiny-Pergola window is divided into two panels. The left panel contains user controls that allow the interactive selection of data range, data time intervals used for display, groups, plot-types and rendering of a line plot for the group average (penultimate track in right panel). The left browser panel displays data in as many plots as selected by the user in the left panel. At the top, the “GenomeAxisTrack” object, used in Gviz to reference the genomic region displayed, has been adapted as a time axis showing the interval of the longitudinal data displayed. Below the time axis, the interval plot displays discrete time intervals such as meals. Pergola estimates accumulated or mean values over defined time windows. This continuous-valued data is displayed by Shiny-Pergola as a heatmap, as shown below the raw meals. Next, time windows values are grouped and their mean is represented as a line plot to help the user identify group-wise patterns. The color code identifying groups (in this case food type) is maintained in these three first plots and is displayed in the legend at the bottom. Gray represents control mice fed only with standard chow while orange represents mice offered exclusively high-fat food after the first week, respectively. The last track can be used to display metadata, in this case the day/night cycle by white and black bars, respectively. Finally, Shiny-Pergola enables the user to download the figures. Note also the heatmap color scale legend on the bottom. Related to Figure 4.

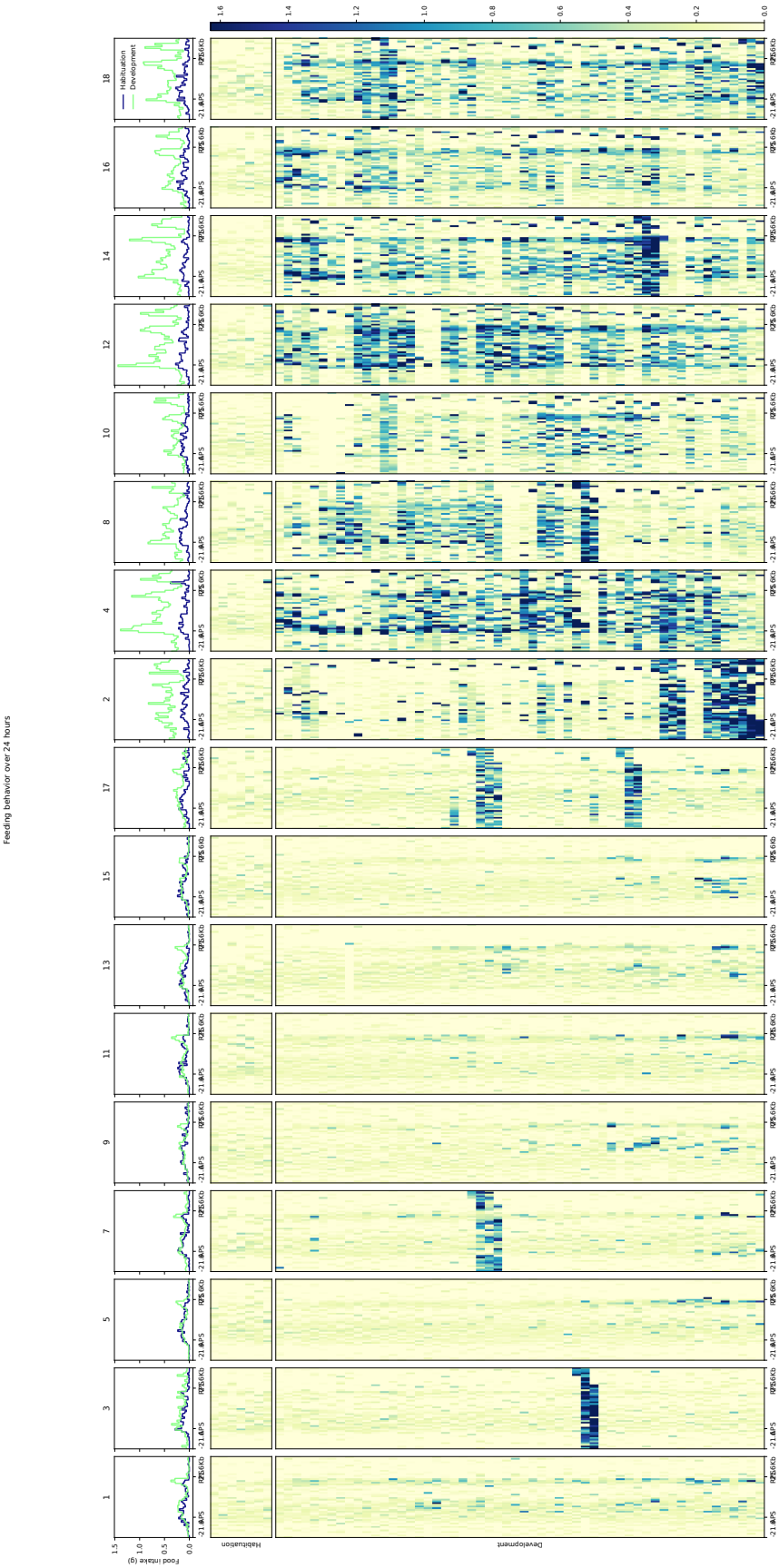

Feeding behavior over 24 hours
Food intake (g)
Habituation
Development
Time (s)

Figure S3 - Individual actograms of mice feeding activity created using deepTools. The figure represents the mouse feeding activity of each individual mouse in the columns (odd IDs on the top correspond to controls and even IDs to high-fat diet mice). In all plots the x axis depicts a 24 hours cycle, each individual time point corresponds to a 30 minutes time window. The region defined within the two ticks encloses the active phase (active phase start: APS and resting phase start: RPS). For each individual mouse, daily activity patterns are summarized on the top. The y axis represents the average food intake (grams) and the two lines display the average activity during the habituation (blue) and obesity development phase (green). On the middle and bottom panel, daily feeding activity is stacked in chronological order. Hence, the middle panel corresponds to 24 hours circadian periods of the habituation phase while the bottom panel summarizes the same cycles but of the obesity development phase. On the heatmaps, color intensities represent the amount of food consumed during the 30 minutes time window.

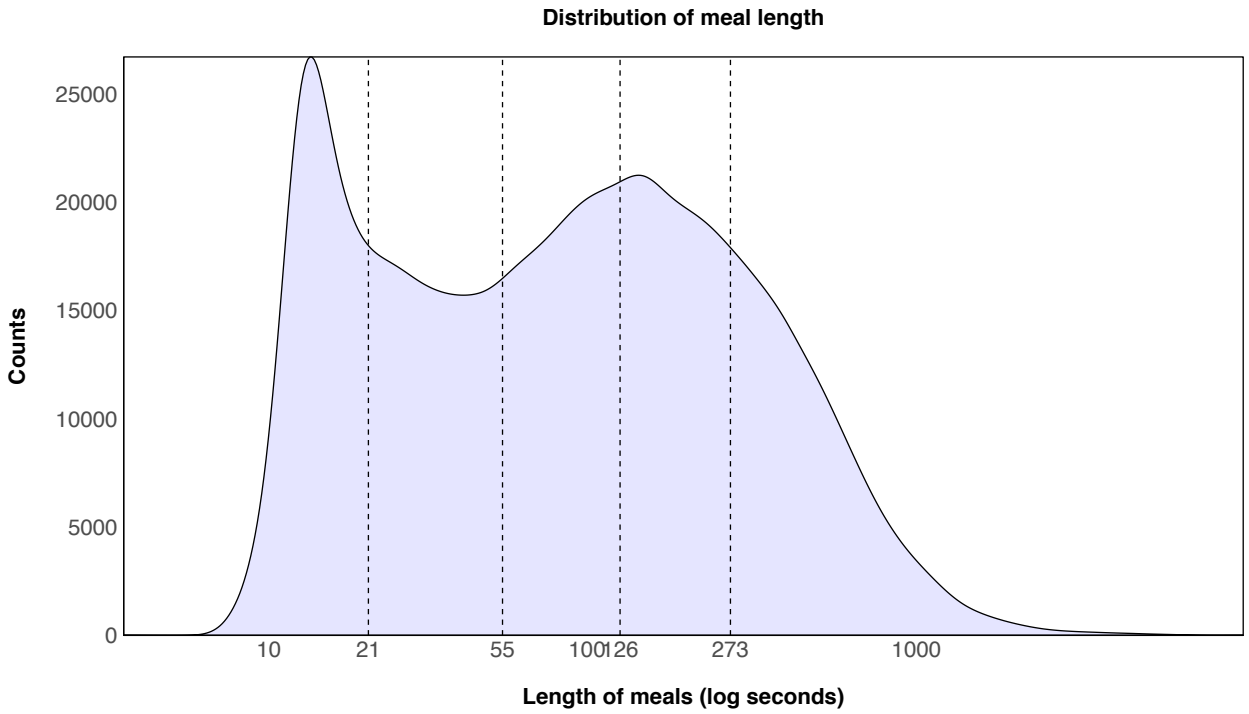


Figure S4 - Distribution of mouse meals by duration distributed in equally sized bins. Plot showing the distribution of feeding bouts by duration. Its x coordinate represents meal length on a log scale and its y coordinate counts. We divided the distribution in 5 bins of an equal size in terms of counts for the annotation with chromHMM.

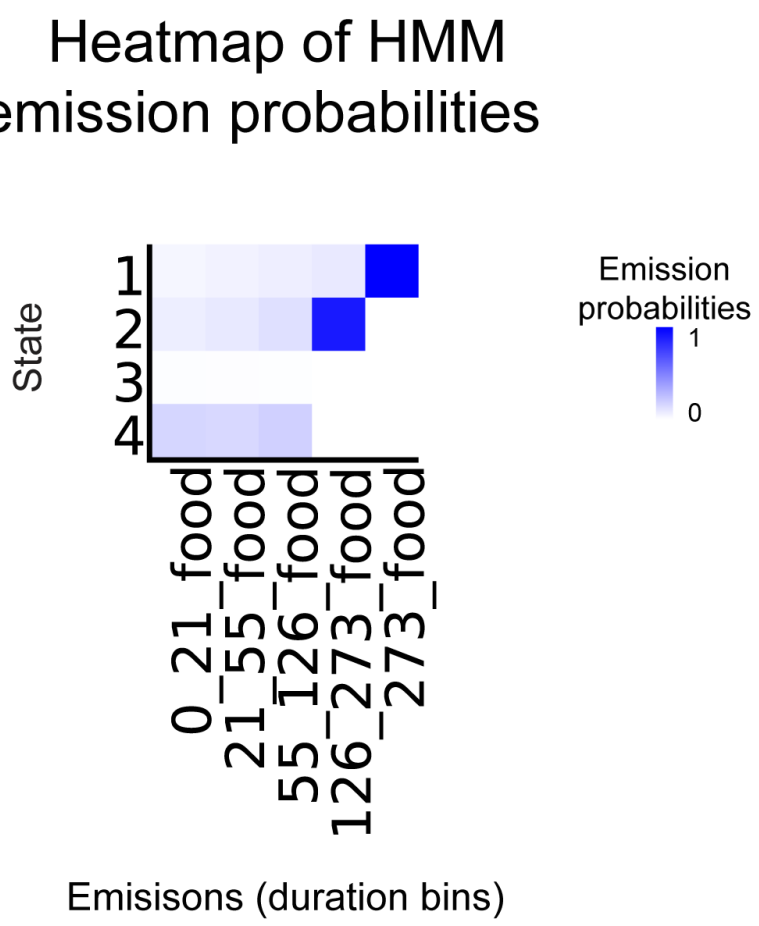


Figure S5 - Heatmap of HMM emission probabilities. Heatmap of emission probabilities as output by chromHMM. On the x axis, duration bins of feeding intervals are shown. The y axis depicts the four states of the HMM. Color intensities on the heatmap represent the probability of emitting a given duration bin with higher blue intensities corresponding to higher probability. States were assigned the following labels in the main text: 1=long meals, 2=regular meals, 3=inactive and 4=short meals.

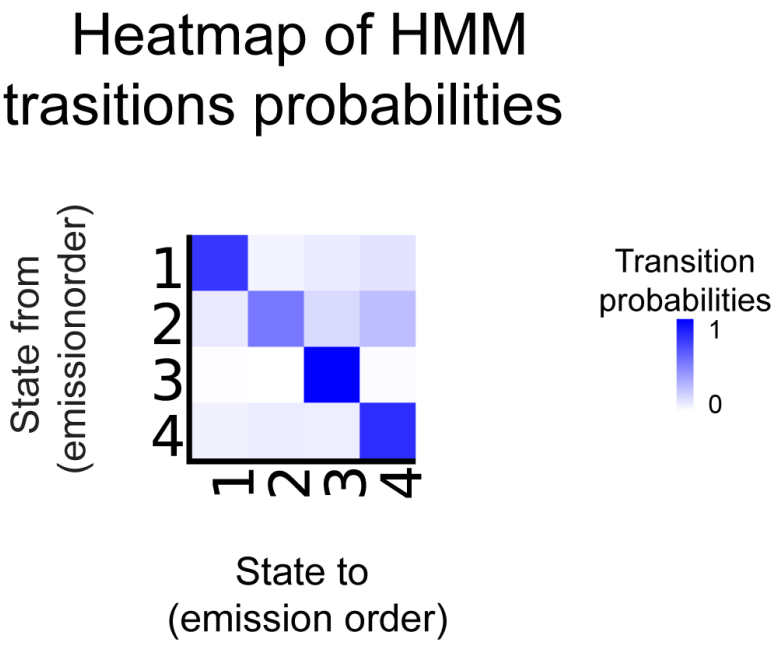


Figure S6 - Heatmap of HMM transition probabilities. Heatmap of transition probabilities as output by chromHMM. On the x axis and y axis the 4 “feeding” states are shown. Each square depicts the probability of moving from a state represented in one of the axes towards the state depicted on the other axis. Color intensities on the heatmap represent the probability of these transitions with higher blue intensities corresponding to higher probabilities. States were assigned the following labels in the main text: 1=long meals, 2=regular meals, 3=inactive and 4=short meals.

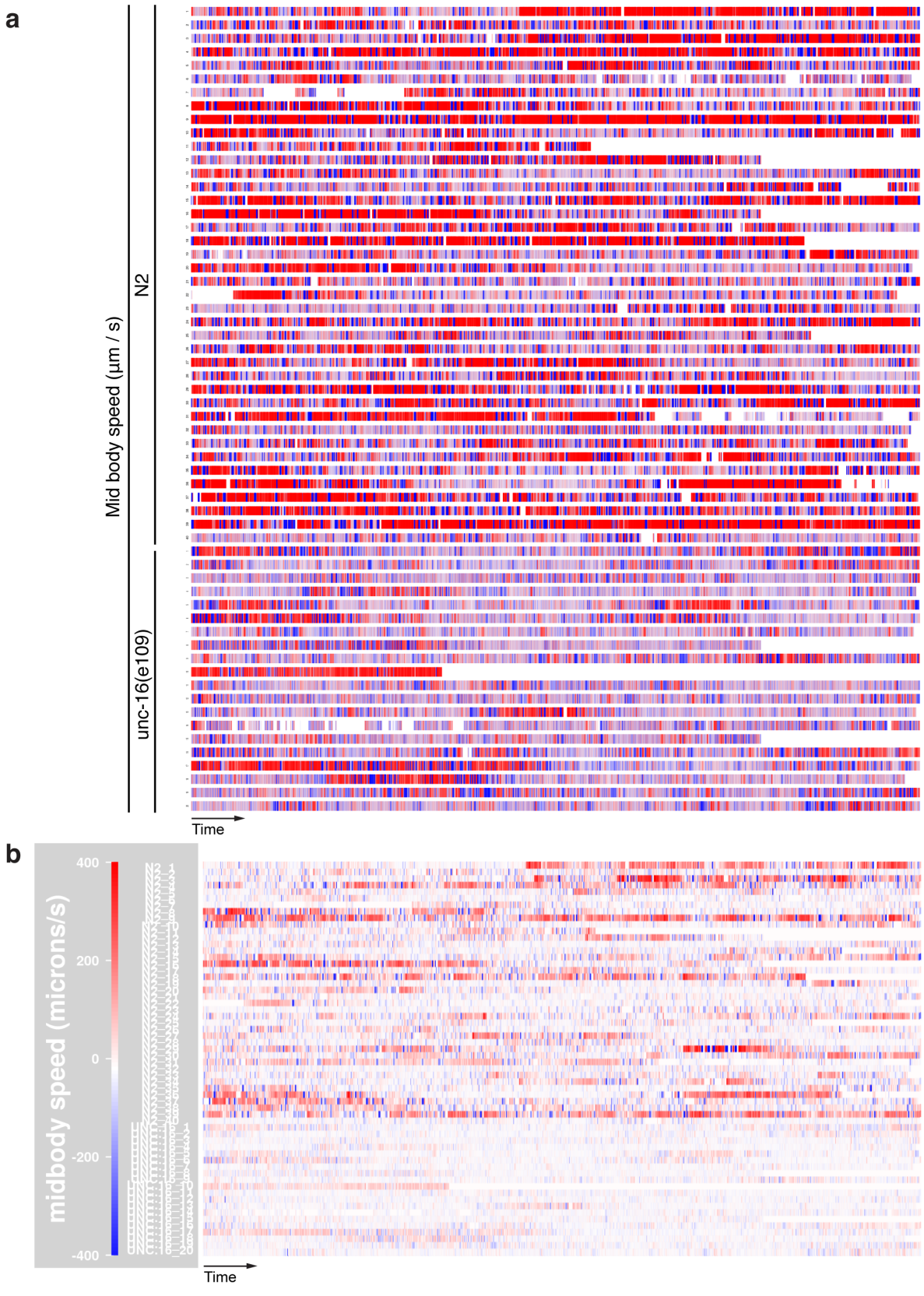

a
Mid body speed (µm / s)
N2
unc-16(e109)
Time
b
midbody speed (microns/s)
400
200
0
-200
-400
Time

Figure S7 - Visualization of worm behavioral trajectories using Sushi and Gviz. Tracks corresponding to *C. elegans* unc-16 mutants (20 tracks) and controls (40 tracks) along time, where blue and red indicate the speed of the forward and backward movements respectively. **(a)** represented using Suhi and **(b)** Gviz. Related to Figure 1.

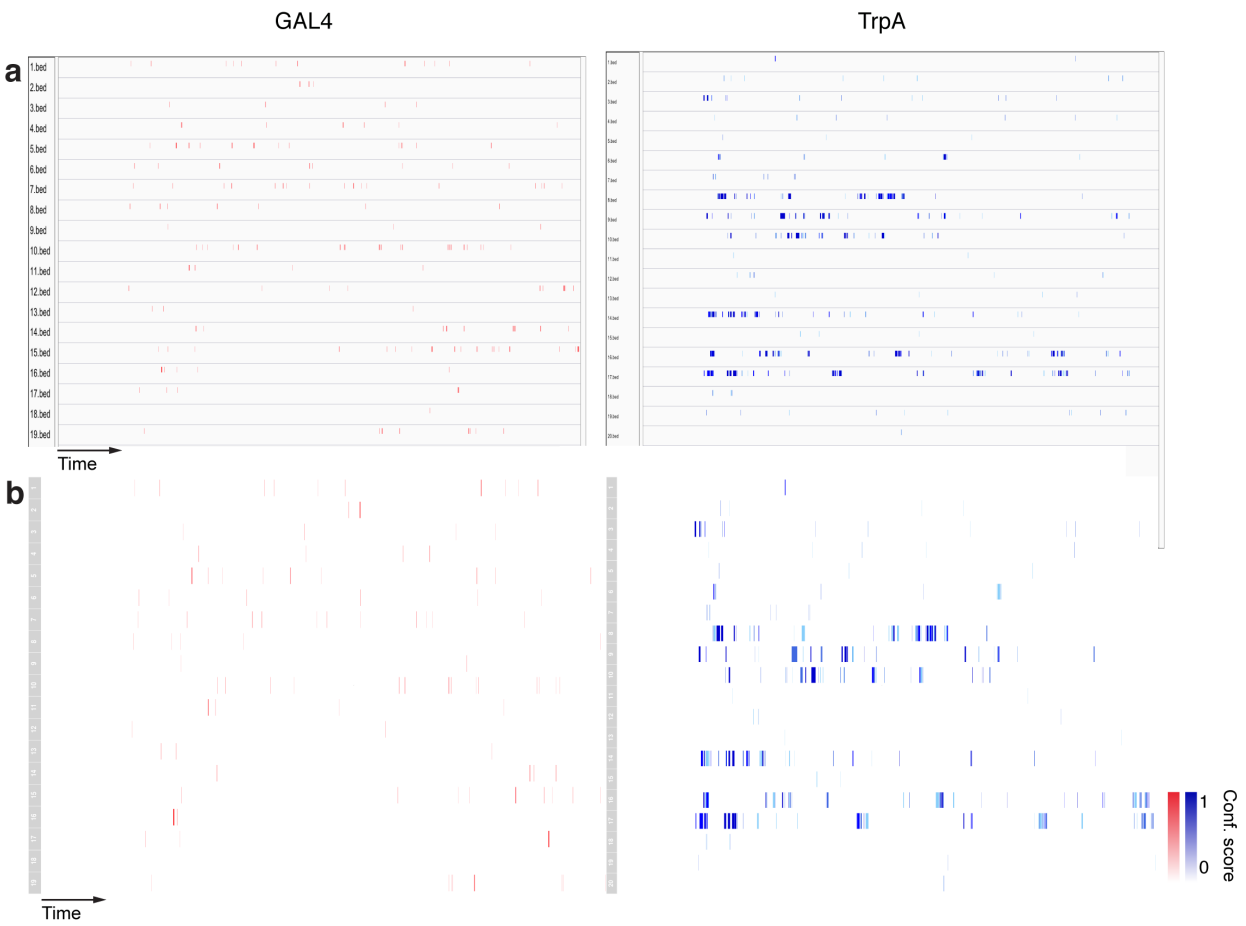


Figure S8 - Visualization of drosophila behavioral trajectories using IGV and Gviz. Tracks representing chase behavior annotated in GAL4 control flies (red, left panels) and TrpA chase-prone strain (blue, right panels). Data has been visualized with **(a)** IGV and **(b)** Gviz. In both cases, color intensity is proportional to the confidence score of the JAABA behavioral classifier. Related to Figure 2.

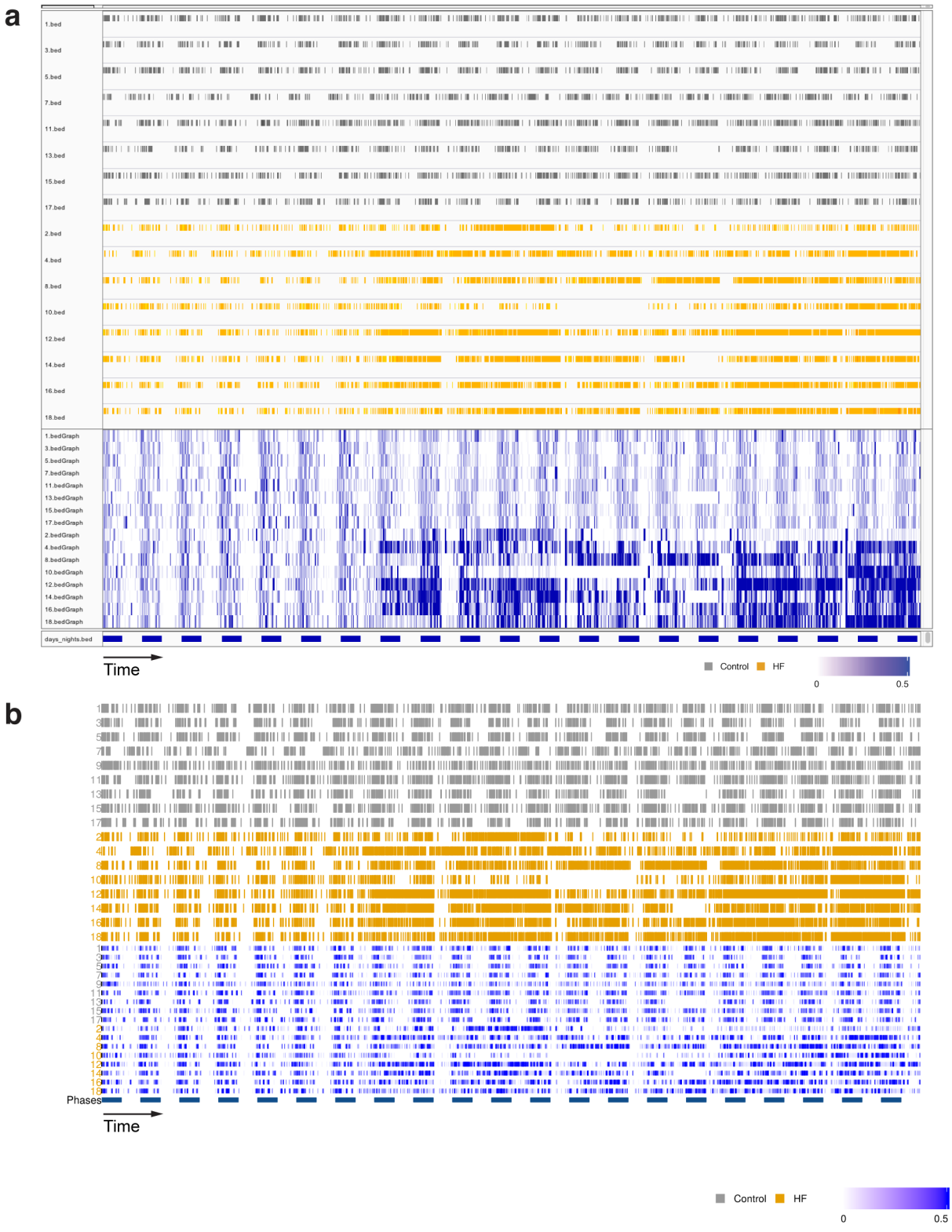

a
1.bed
3.bed
5.bed
7.bed
11.bed
13.bed
15.bed
17.bed
2.bed
4.bed
8.bed
10.bed
12.bed
14.bed
16.bed
18.bed
1.bedGraph
3.bedGraph
5.bedGraph
7.bedGraph
11.bedGraph
13.bedGraph
15.bedGraph
17.bedGraph
2.bedGraph
4.bedGraph
8.bedGraph
10.bedGraph
12.bedGraph
14.bedGraph
16.bedGraph
18.bedGraph
days_nights.bed
Time
Control
HF
0
0.5
b
Phases
Time
Control
HF
0
0.5

Figure S9 - Visualization of mouse behavioral trajectories using IGV and Sushi. Mice feeding behavior corresponding to the first 3 weeks of experiment visualized with **(a)** IGV and **(b)** Sushi. In both cases top panel displays raw meals. Each meal is represented as a bar with its length corresponding to duration. Bottom panel displays the accumulated values of food during time windows of 30 minutes as a heatmap, with high values depicted as dark blue (maximum set to 0.5 grams) and low values depicted as light blue (gray: control group, orange: High-fat diet group). Related to Figure 3.

Supplemental Tables

! Mapping of behavioral fields into pergola ontology terms

! comments

behavioral_file:phase_exp	>	*pergola:chrom*
behavioral_file:individual_id	>	*pergola:track*
behavioral_file:event_start	>	*pergola:start*
behavioral_file:event_end	>	*pergola:end*
behavioral_file:type_of_event	>	*pergola:data_types*
behavioral_file:value	>	*pergola:data_value*

Table S1 - Generic conversion file. To establish the correspondence between the fields of the input data and Pergola ontology, Pergola uses the “External Mapping File Format” from the Gene Ontology Consortium. Each individual syntax line consists of a left part and a right part connected by a “>” symbol. The left part indicates the external source (in our case the input file) and, separated by colon, the term identifying the column in the input file. The right part indicates the tag identifying the Pergola ontology and, separated by colon, the term describing the output column in Pergola ontology.

Term name	Mandatory	Definition
start	Yes	Refers to start time points of each interval of the original data. If “chrom_end” is not set all “chrom_start” should be equidistant and intervals will be set to the delta between time points.
data_value	Yes	Refers to associated values considered for the representation of data.
end	No	Refers to the end of each time interval.
track	No	Refers to each of the experimental entities present in the file.
data_types	No	Refers to each of the different features annotated in the file.
phase	No	Refers to different phases of the experiment.
dummy	no	All additional fields in the original input data not used by Pergola

Table S2 - Pergola controlled vocabulary. Pergola ontology terms are used to describe the types of information that are contained in the longitudinal input files. By tagging each of the input fields with this vocabulary, Pergola can interpret the information from the input files and process the data into the suitable genomic format in a consistent manner.

Transparent Methods

Pergola software

Overview. Pergola encodes time series data into files formatted according to the most common genomics formats. In order to achieve this task, it requires as input data a file or a series of files (Figure 1a) containing the raw sequential data to be converted (typically a spreadsheet-compatible file in CSV or XLSX format) and a mapping file (Table S1). As the name implies, the mapping file declares the equivalences between the terms in the sequential file and an ontology (table S2) in which relevant longitudinal concepts are present. In this manner, Pergola establishes what type of information contains each input file column (e.g. unit of time in the user file is declared as a unit of time in Pergola and internally mapped onto nucleotide positions when processing). Once the mapping file is ready, any user data file can be processed and reformatted into widely used genomic formats. Formats such as GFF or BED constitute a perfect scaffold for discrete time series of behavior (Figure 1b), while bedGraph or bigWig formats are perfectly suited to encode continuous time series of behavior in (Figure 1c). Additionally, during data conversion Pergola enables to invoke some processing utilities, such as grouping or binning of the data (see Pergola basic utilities section below). The output files can then be handled for further analysis and visualization (Figure 1d) using popular third party genomics tools such as desktop Genome Browsers (Robinson and Thorvaldsdóttir, 2011), BEDtools (Quinlan and Hall, 2010), Bioconductor packages (Gentleman et al., 2004), deepTools (Ramírez et al., 2016) or chromHMM (Ernst and Kellis, 2012).

Input data. Pergola currently accepts four input data formats: (i) comma-separated values (CSV) files, a common export format for tabular data compatible with commercial-spreadsheet

software and most behavior-tracking applications. (ii) XLSX files, default Microsoft Excel format used by spreadsheet applications. (iii) ISA-Tab format, a wrapper format for CSV data generated in high-throughput studies (Rocca-Serra et al., 2010). The ISA-Tab format seeks to provide a standard for the maintenance of data in complex experiments along with the necessary metadata for its interpretation. (iv) JAABA output files in Matlab format (Mathworks). In all cases the minimum input is a two-column file containing the timestamps and the variable measured.

Mapping file. The mapping file is a two-column text file used to declare the equivalences between input data terms (e.g. the column names or its index in the excel file) and Pergola controlled terms (Table S2). As this file declares a mapping between two ontologies, its format was designed to follow the Gene Ontology Consortium community (Ashburner et al., 2000) standards specifications (http://geneontology.org/page/external-mapping-file-format). To illustrate how the mapping works, consider an input file such as the one of our toy example (Figure 1a). In the corresponding mapping file (Table S1), initial timestamps in input data (column “event_start”) are declared to be mapped onto “start” interval positions of the Pergola ontology. As a result, every value in the first column of the original data will be treated as the start coordinate of a behavioral feature in the genomic format.

behavioral_file:event_start > pergola:start

In the assignment above, “behavioral_file“ is the alias name of the input file and “event_start“ the name of the column containing the timestamps. In the part of the assignment after the “>“ sign, “pergola“ indicates the name of the source ontology and “start“ the selected term in that ontology.

Discrete output formats. One of the most common types of behavioral data consists of a sequence of behavioral bouts, defined as intervals of time within which the behavior has taken place (e.g. meals). These bouts can be qualitatively associated with quantities recorded along a temporal trajectory (Figure 1b). Following the meal example, these quantities will account for the food intake consumed during the interval. Formats used to encode genomic annotations or to describe sequence features provide a perfect scaffold to define such behavioral trajectories. Pergola supports two of these genomic formats: the Browser Extensible Data (BED) (Kent et al., 2002) or the General Feature Format (GFF) (https://github.com/The-Sequence-Ontology/Specifications/blob/master/gff3.md).

Continuous output formats. In a genome, annotation can be in the form of a discrete interval defined both by a start and end position (e.g. coding or noncoding) or in the form of a continuous score covering a genomic sequence (e.g. level of expression). The bedGraph format encodes precisely this type of continuous scores. Being able to annotate quantitative data is essential in order to maintain without loss the information associated with discrete behavioral recordings (i.e. speed during a whole trajectory, or only within forward movement intervals, or amount of food intake during a time window). Pergola adopted the bedGraph format to carry out lossless conversion of any continuous behavioral annotation (Figures 1c). In the example shown in Figure 1, the scores are produced by the binning utility of Pergola, however, it is also possible to directly transform continuous recordings, as we have shown in the speed trajectories analysis of the *C. elegans* data set (Figure 1a).

Other formats. Indexed binary formats such as bigBED (discrete) or bigWig (continuous), among others, facilitate the efficient storage of big chunks of data. Although not currently supported by Pergola, the conversion between diverse genomic data types becomes trivial thanks to the many tools available for that purpose (Kent et al., 2010). It is worth mentioning that

some tools require specific file formats and, to apply some of the deepTools functions in our computations, e.g., we first needed to transform bedGraph files to bigWig. Hence, although this format is not directly adopted in our library to avoid software dependencies, our server can convert data directly to bigWig format.

Implementation and availability. Pergola is implemented in Python and released under the GNU general public license version 3. Latest version, source code, documentation and issue tracking are available at http://cbcrg.github.io/pergola/. The package is also available for installation on the Python Package Index (PyPI) at https://pypi.python.org/pypi/pergola. Pergola releases are provided as Docker containers in a public Docker Hub account (https://hub.docker.com/r/pergola/pergola/). The version used for the analysis in this paper (0.1.5) is accessible under the "pergola/pergola:0.1.5" tag.

Pergola basic utilities. Pergola allows to apply some basic operations on the time series data before its conversion to genomic formats. Basic functionalities include:

- Group several tracks together by ID or data type.
- Bin the data by windows of an optional time step.
- Summarize the mean information by window.
- Obtain BED files of consecutive equidistant segments (e.g. 12 hours to annotate light and dark phase of a circadian rhythm).
- Parse the data from an initial to a final time point with the rest of the records being ignored.
- Reference all timestamps relative to the initial time point of the data. This way the initial point becomes time zero.

A more thorough description of these and other functionalities can be found in the Pergola documentation (http://cbcrg.github.io/pergola/).

Web server. To make Pergola accessible to a wider research community, we developed a web server available at http://pergola.crg.eu/. Among other things, the server provides a user-friendly interface and an easy way to share data visualizations across laboratories. Server visualization utilities have been implemented using the lightweight html5 version of the IGV, available in the repository https://github.com/igvteam/igv.js/tree/master.

Third-Party Genomic Analysis Tools

Once reformatted by Pergola, behavioral data can be analyzed and visualized using a wealth of genomic tools. In the context of this work we used the following software and algorithms to perform the analysis shown in the result section:

BEDTools is a software suite frequently used to query genomic data (Quinlan and Hall, 2010). It allows arithmetic operations to be performed on genomic data such as intersect, complement, merge or count, among others. These operations are also relevant for longitudinal behavioral data. For instance, one may use it to programmatically extract a subset of longitudinal recordings fulfilling a given condition (chasing in fly, motion state in worm, feeding bout in mice, etc…). We also carried out some of our analysis using **Pybedtools** (Dale et al., 2011), a Python wrapper that extends BEDTools and allows combining BEDTools utilities with Pergola within the same Python scripts.

Bioconductor packages. Bioconductor provides multiple R packages for the analysis of genomic data. We explored the application of several of these packages to the analysis of longitudinal behavioral data. The **Rtracklayer** (Lawrence et al., 2009) package was used to

import behavioral data in various genomic file formats into R. We also used the **GenomicRanges** (Lawrence et al., 2013) package to manipulate the data and the **Sushi** (Phanstiel et al., 2014) and **Gviz** (Hahne and Ivanek, 2016) packages for visualization.

IGV adapted version. We adapted IGV (Robinson and Thorvaldsdóttir, 2011), a popular desktop genome browser, to display temporal measurements instead of genomic features. For that purpose, we forked the GitHub IGV repository https://github.com/igvteam/igv and modified it accordingly https://github.com/JoseEspinosa/IBB.git. While this adapted version has been used to produce the rendering shown in Figure 2a, the native application could generate a similar visualization since our modifications involved only cosmetic improvements (e.g. display time units instead of nucleotide positions).

kentUtils is a collection of tools maintained by the UCSC Genome Browser that allows operations on genomic binary formats (Kent et al., 2010). We used them to convert Mice bedGraph files to bigWig.

WiggleTools consists of a compendium of statistical tools developed for the analysis of large genomic data sets (Zerbino et al., 2014) to obtain the groupwise mean of all the mouse feeding trajectories. This is a necessary step prior to the creation of the group actograms using deepTools.

DeepTools is a suite of genomic tools for the analysis of high-throughput sequencing data (Ramírez et al., 2016). Using its utilities, a principal component analysis (PCA) was performed on the worm data set. We further obtained mouse feeding activity profiles akin to profiles used

to depict the enrichment or depletion of genomic features flanking promoters or Transcription Start Sites.

chromHMM is an implementation of a multivariate hidden Markov model that has been extensively applied to model annotations of chromatin states (Ernst and Kellis, 2012). The input of this algorithm consist in a series of aligned reads that contain information about the epigenetics marks all along the genome. The algorithm integrates this information and uses it to annotate chromatin states by applying a hidden Markov model (HMM). By binning the feeding activity by size in BED files, we could apply this tool to annotate mice behavioral trajectories.

Galaxy is a scientific workflow platform to ease handling, analysis and publication of experimental and computational data throughout a web interface. Many popular Bioinformatics tools provide plugins for this system. We used the main Galaxy instance (Afgan et al., 2018) (http://usegalaxy.org/) with available wrappers for default deepTools and BEDTools to reproduce the example analysis on mice within this framework.

Shiny-Pergola. Based on the Shiny R library (http://shiny.rstudio.com/), we developed Shiny-pergola, a web based application for the interactive visualization of longitudinal behavioral data. The application integrates the Gviz, GenomicRanges and rtracklayer Bioconductor packages to render behavioral annotations and continuous behavioral data in BED and bedGraph formats and is available as a free open-source package from https://github.com/JoseEspinosa/shiny-pergola-docker. We generated the graphics from the longitudinal visualization of mouse feeding data (Figures 4 and S2) using Shiny-pergola.

Reproducibility

Computational repeatability. All the analyses carried out here are provided following the FAIR principle (Findable, Accessible, Interoperable and Reusable) (Wilkinson et al., 2016). We therefore followed the distribution model recently proposed in Di Tommaso et al. Data has been uploaded to the Zenodo research repository, all the analysis code has been open-sourced on GitHub (https://github.com/cbcrg/pergola-reproduce) and the analysis pipelines themselves were implemented using Nextflow (Di Tommaso et al., 2017) version 0.30.2 downloaded from https://github.com/nextflow-io/nextflow. The software and dependencies used in all the analysis were wrapped up in a single Docker image that is publicly available at https://hub.docker.com/r/pergola/pergola-reproduce/ under the "paper" tag, thus making it possible to run the analysis avoiding the installation of pipeline dependencies in the target environment. Based on r-base (R version 3.3.3), the image is referenced in each pipeline configuration file using the SHA256 identifier. This unique hash identifier guarantees the permanent access to the exactly same image and thus preserves the immutability of the software.

Datasets. Three publicly available datasets were selected and processed through the Pergola framework. These datasets were selected because they contained analysis based on longitudinal behavioral recordings coming from three different model systems. One part of the analyses was focused on establishing the reproducibility of the original observations in the Pergola framework, while additional analyses were meant to show how further insights can be obtained by applying statistical techniques implemented in genomic software tools. The following sections describe the analysis corresponding to each of the datasets.

C. elegans analysis

***C. elegans* dataset.** We obtained worm motor behaviors from the "*C. elegans* behavioral database" (Yemini et al., 2013). Specifically, we used data from a mutant unc-16 strain (n=20), a mutation affecting locomotion, and the N2 control strain (n=40). Each individual recording was available in an HDF5-formatted file (Hierarchical Data Format Version 5). Files consisted of a time series of raw phenotypic features frame by frame plus all the experimental metadata (see Yemini et al. for a detailed description). The original database described in Yemini et al. (http://wormbehavior.mrc-lmb.cam.ac.uk/) is no longer maintained since it has been recently migrated to http://movement.openworm.org/. The data was obtained from this new repository and has been uploaded to Zenodo (see Data and code availability section).

Computational analysis. Using Pergola, we extracted motion states (worm moving forward, backward or paused) into three BED files, with each BED file containing all the intervals of the time a given individual spends in the given motion state. Conversely, mid body speed of individual worms was formatted to bedGraph files where each coordinate represented a single time frame. Applying the BEDTools intersect function, we extracted the parts of the speed trajectory corresponding to each of the motion states. As illustrated in Figure 3, bedGraph files containing speed annotations of a single worm were intersected with the BED file representing those intervals of the animal trajectory assigned as moving forward. The resulting files were grouped by motion state and were used to compare the body speed densities of each worm strain (as depicted in Figure 2) by means of an R script. Apart from this analysis, we used the "multiBigwigSummary" and "PCAplot" functions of deepTools to cluster the longitudinal behavioral trajectories from the *C.elegans* data set using a principal component analysis (PCA). Complete worm trajectories in bedGraph files were formatted as bigWig files to enable the

computation of the PCA plot. To obtain the variables provided to the PCA, the bigWig files have to be aggregated into bins. Here, deepTools was used to transform each worm trajectory into 10 bins of equal length (2900 frames).

Visualization. IGV was used to display worm bedGraph files corresponding to the mid body speed as a heatmap (Figure 1a). The description of how to visualize worm data on IGV is available under the following link: https://github.com/cbcrg/celegans-pergola-reproduce.

Versions. BEDTools (v2.26.0) was installed using a Debian package manager. Sushi (v1.12.0) and Gviz (v1.19.2) libraries were installed using Bioconductor (v3.4). The Gviz library adapted to display time instead of genomics units is available from the GitHub repository under the following link https://github.com/JoseEspinosa/Gviz/tree/fps. Version 3.1.1 of deepTools was installed using PyPI.

Commands. The Nextflow pipeline can be run cutting and pasting the following command:

```
$ nextflow run celegans-pergola-reproduce.nf \
      --strain1_trackings 'data/unc_16/*.hdf5' \
      --strain2_trackings 'data/N2/*.hdf5' \
      --mappings_speed 'data/mappings/worms_speed2p.txt' \
      --mappings_bed 'data/mappings/bed2pergola.txt' \
      --mappings_motion data/mappings/worms_motion2p.txt \
      -with-docker
```

Data and code availability. The input dataset has been uploaded to Zenodo and can be accessed under the following DOI: https://doi.org/10.5281/zenodo.1101067. Pipeline source

code together with results have been archived on Zenodo as well, and are accessible with the following DOI: https://doi.org/10.5281/zenodo.1068506. The pipeline together with detailed instructions on how to replicate the analysis can also be found in the GitHub repository at this link: https://github.com/cbcrg/celegans-pergola-reproduce.

Fly analysis

Fly JAABA dataset. *Drosophila melanogaster* data was downloaded from: https://sourceforge.net/projects/jaaba/files/Sample Data/sampledata_v0.1.zip/download. We processed the original data consisting of motor trajectories derived from video-recordings spanning 1000 seconds using JAABA to annotate chasing behavior. The dataset contains a group of 20 *GAL4* fruit flies from a TrpA activation screen and a control group of 19 pBDPGAL4U flies. The *GAL4* line shows an increased propensity for chasing. Motor trajectories were originally obtained with Ctrax (Branson et al., 2009) and used in the JAABA publication (Kabra et al., 2013). Data has been made available on Zenodo (see Data and code availability section).

Computational analysis. We used JAABA to identify chasing behavioral bouts. The resulting annotations derived from fly trajectories were processed by a Python script which combined Pergola and Pybedtools. Using Pergola, we converted the annotations into Pybedtools BED objects and obtained a subset of each behavioral trajectory annotated (i.e. “covered” in genomic jargon) as chase behavior by applying the “genome_coverage” utility. The resulting “coverages” were then used to compare the differences between the fraction of time the two fly lines spent chasing (Figure 4b). Finally, we applied a volcano plot to identify which variables derived from the fly posture tracking accounted for the highest and most significant differences between

periods labeled as chase behavior by JAABA and unlabeled periods. Inspiring ourselves in differential analysis of gene expression data, we considered labeled and unlabeled regions as the two experimental conditions. Therefore, for each variable from the posture tracks we collected all the measurements within chase-labeled and unlabeled periods. This was done across all individuals regardless of strain. We then compared the means between labeled and unlabeled periods to obtain a fold change and determined the significance of this change using a t-test. As commonly done, log2 of the fold change is then plotted against -log10 of the p-value, to display both amount of change between conditions and its significance for each of the variables.

Visualization. Once converted to BED files, JAABA chase annotations were visualized using the Sushi R Bioconductor package (Figure 3a). Pergola can assign a color gradient to the values associated with BED files intervals. In this case, we used this feature to depict the JAABA confidence score for the behavioral classifier (the higher the confidence the darker the interval is shaded).

Versions. Pybedtools (v0.7.10) were installed using pip install. Sushi (v1.12.0) and Gviz (v1.19.2) libraries were installed using Bioconductor (v3.4). The Gviz library was adapted to display time instead of genomics units and is available from the GitHub repository under the following link https://github.com/JoseEspinosa/Gviz/tree/fps.

Commands. The fly analysis can be run using the command below:

```
$ nextflow run melanogaster-pergola-reproduce.nf \
      --scores='data/scores/scores_chase_*.mat' \
      --var_dir='data/perframe_*' \
```

```
--variables="all" \
--mappings='data/mappings/jaaba2pergola.txt' \
-with-docker
```

Data and code availability. The input dataset has been uploaded to Zenodo and can be accessed under the following DOI: http://doi.org/10.5281/zenodo.1067835. Pipeline source code together with results have been archived on Zenodo as well and are accessible under the following DOI: https://doi.org/10.5281/zenodo.1068334. The instructions to replicate the fly analysis along with links to the input datasets and images can be found on GitHub under this link https://github.com/cbcrg/melanogaster-pergola-reproduce.

Mouse analysis

Mouse dataset. The mouse dataset consists of longitudinal recordings of behavior. Recordings were compiled over 9 weeks on 17 C57BL6/J male mice, 9 assigned to the control group and 8 assigned to the high-fat (HF) diet group. The animals were tracked individually by the PHECOMP food and drink monitoring system as published by Espinosa-Carrasco et al. Mice from the two groups were provided with *ad libitum* access to a standard chow within the first week of the experiment to define the baseline behavior and to allow the habituation of the animals to the experimental conditions (habituation phase). After this period, the HF mice group was put on an exclusive diet of a high-fat content chow while controls remained having access to only standard chow. This diet regime was maintained during 8 weeks, a period of time within which HF mice developed obesity (development phase). During the whole experiment, feeding behavior (including intake, and duration of each feeding bout) was tracked by the system (see

Espinosa-Carrasco et al. for more details). The mouse dataset has been uploaded to Zenodo (see Data and code availability section).

Computational analysis. Data from the Phecomp system was processed into BED and bedGraph files using Pergola. BED files were employed to encode the raw feeding bouts, the day/night cycles and the experimental phases. Individual food intakes were aggregated over time windows of 30 minutes using Pergola, and the resulting scores were stored in bedGraph files. Using Pybedtools, we summarized some feeding behavioral measures (average intake during a meal, number of meals, its duration, mean eating rate and the separation between two meals or average intermeal duration) and used them to quantify the disruption of these measures in the HF mice when compared to controls. We further represented them in a heatmap using R scripts (Figure 5). To create the actograms-like plots, bedGraph files containing the 30 minutes scores were converted into bigWig files using “bedGraphToBigWig”. The further commands to produce the actograms are detailed in the section “Code to reproduce deepTools circadian profiles” below. Finally, the BED files corresponding to the raw feeding bouts were used to model feeding activity trajectories using chromHMM. We first divided the distribution of feeding activity bouts by length in 5 bins of the same sample size. Since the overall length distribution was bimodal (Figure S4), the choice of quintiles resulted in the assignment of two bins for the part of the distribution peaking around 10 seconds, two more bins for the part of the data peaking around 120 seconds and a remaining bin for the tail of the distribution. In this way, bins were expected to represent short, regular and long meals, respectively. We then created a design file as described in the protocol by (Ernst and Kellis, 2017). In short, we considered each mouse as a different sample (“cell type” in the original context) and the five different duration bins (“epigenomic marks” as described in the protocol) as the observed variables. The algorithm was then applied to the binned BED files as described in its documentation to annotate 4 “behavioral states" (as opposed to the original “chromatin

states“). The resulting HMM annotations were used to compare the fraction of time spent in each of the states after the introduction of the high-fat food (obesity development phase) with respect to the basal condition (habituation phase) in the high-fat group. In the same manner, the behavior of controls was compared between these two phases, even though this group was not subjected to any dietary change. For this comparison we used a spie chart, a representation that allows the contrast of two pie charts by setting the angles of the slices by one of the conditions and the areas by a comparison with the other one. Here, the angles show the fraction of time within a state during the habituation and the area is determined by a state’s occurrence ratio between development and habituation.

Code to reproduce deepTools circadian profiles

We believe that the profiles of mice feeding activity (Figure 7 and S3) nicely exemplify how easy it can be to obtain an insightful visualization of behavioral time series, which otherwise would require the implementation of a big chunk of code. Here, we show the necessary commands needed to reproduce Figure S3 once the data has been converted to genomic formats with Pergola. First, deepTools calculates a score matrix of the regions of interest (in this case dark/active phases) using the following command:

```
$ computeMatrix scale-regions \
      -S $list_of_bigWig_ctrl $list_of_bigWig_hf \
      -R dark_habituation.bed dark_development.bed \
      --beforeRegionStartLength 21600 \
      --regionBodyLength 43200 \
      --afterRegionStartLength 21600 \
```

```
        --skipZeros -out matrix.mat.gz
```

In short, this command takes the list of bigWig files (here control and high-fat mice represented as a bash variable) and the regions of interest encoded as BED file (in this case all the dark/active periods of both the habituation and development phase). The “regionBodyLength” option sets the length of the central region of the heatmap (between the ticks, see Figure S3). The regions represented upstream and downstream of this central region can be provided using “beforeRegionStartLength” and “afterRegionStartLength” parameters. In our case, we set them to 21600 seconds corresponding to the 6 hours previous to the start of the active phase and the 6 hours after it, hence covering the whole daily 24 hour period. We then obtain the scores to be used for the plotting of the heatmaps and profiles in a matrix using the following command:

```
$ computeMatrix -m matrix.mat.gz \
      -out heatmap_actogram_like.pdf \
      -z Habituation Development \
      -T "Feeding behavior over 24 hours" \
      --startLabel APS \
      --endLabel RPS \
      --xAxisLabel "Time (s)" \
      --yAxisLabel "Food intake (g)" \
      --sortRegions no \
      --plotFileFormat pdf \
      --colorMap YlGnBu
```

In this command, the "-m" flag is the essential piece of information since it is used to input the score matrix to the function. The rest of options are just aesthetic parameters, except for the "sortRegions" option that avoids that the 24 hour periods are stacked by the intensity of the signal instead of chronological order. Remarkably, once data is processed by Pergola, only two commands are enough to reproduce the figure.

Since the main Galaxy instance (https://usegalaxy.org/) includes deepTools as well as other text and BED processing applications, it is actually quite straightforward to produce this same figure just by using an Internet connection and without having to recur to the computer terminal.

A Galaxy description page where sample files and program instructions for this example are embedded under https://usegalaxy.org/u/toniher/p/deeptools-pergola. Thanks to this and the Pergola web server (http://pergola.crg.eu), users can reproduce this case easily by themselves. Moreover, they can also use and adapt the same backbone workflow for their own situations. This feature is relevant for researchers not familiar with using the command-line.

Visualization. The mouse dataset has been rendered using Gviz. To read genomic data formats and convert them to Gviz compatible objects, we used the rtracklayer and GenomicRanges Bioconductor packages. The resulting Gviz visualization (Figure 4) shows the BED files corresponding to the meal intervals at the top and a heatmap derived from the bedGraph files depicting intakes by window of time at the bottom. The changes of the feeding behavior associated with the introduction of the high-fat food are evident in this visualization (Figure 4). On the basis of this rendering, we developed the ad-hoc Shiny behavioral browser (Figure S2). For a detailed description of Shiny-Pergola visualization, see the next point.

Shiny-Pergola visualization. To visualize data, we created a dockerized version of Shiny-Pergola that is publicly available at https://hub.docker.com/r/pergola/shiny-pergola/ under the "0.1.1" tag. This Docker image can generate an interactive data visualization when launched

from the same folder where the Nextflow pipeline is run using a configuration file to set group membership using the following command:

```
$ docker run --rm -p 3600:80 -v \
      "$(pwd)":/pergola_data \
      pergola/shiny-
pergola@sha256:6a3668eeb160a04e5fa7853d318be6b0d64a707fa7d8aa98a1468cb3
b27cba86 &
```

Once the previous command is launched, the visualization can be accessed in a web browser on the same computer under http://0.0.0.0:3600.

Versions. Pybedtools (v0.7.10) and deepTools (v3.1.3) were installed using PyPI. The Sushi (v1.12.0) and Gviz (v1.19.2) libraries were installed using Bioconductor (v3.4). The Gviz library that was adapted to display time instead of genomics units is available from the GitHub repository under the following link: https://github.com/JoseEspinosa/Gviz. Version v1.2.2 of WiggleTools was installed from their GitHub repository: https://github.com/Ensembl/WiggleTools . Likewise, version v1.15 of chromHM was installed from the official GitHub repository under this link: https://github.com/jernst98/ChromHMM. Finally, kentUtils version v302.1.0 was obtained from GitHub at https://github.com/ENCODE-DCC/kentUtils

Commands. The pipeline can be executed invoking Nextflow with the following command:

```
$ nextflow run mouse-pergola-reproduce.nf \
      --recordings='data/mouse_recordings/' \
```

```
--mappings='data/mappings/b2p.txt' \
--mappings_bed='data/mappings/bed2pergola.txt' \
--phases='data/phases/exp_phases.csv' \
--mappings_phase='data/mappings/f2g.txt' \
--exp_info='data/mappings/exp_info.txt' \
--tbl_chromHMM="data/chromHMM_files/cellmarkfiletable" \
--n_bins_HMM=5                                                    \
--n_states_HMM=4 \
-with-docker
```

Data and code availability. The input dataset has been uploaded to Zenodo and can be accessed under the following DOI: https://doi.org/10.5281/zenodo.1154827. Pipeline source code together with results have been archived on Zenodo as well and are accessible with the following DOI: https://doi.org/10.5281/zenodo.1068309. A detailed description including all the instructions along with the links to the mouse dataset needed to reproduce this analysis and the visualization can be found in the GitHub repository under the following link: https://github.com/cbcrg/mouse-pergola-reproduce.

Interoperability. Once formatted with Pergola, the resulting data from the three datasets was handled with all of the three visualization procedures, i.e. with IGV, Sushi and Gviz, to show how genomic standards enable data and software interoperability (Figures S7-S9).

Supplemental References

Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G. (2000). Gene Ontology: tool for the unification of biology. Nat. Genet. 25, 25–29.

Branson, K., Robie, A.A., Bender, J., Perona, P., Dickinson, M.H. (2009). High-throughput ethomics in large groups of Drosophila. Nat. Methods 6, 451–457.

Ernst, J., Kellis, M. (2017). Chromatin-state discovery and genome annotation with ChromHMM. Nat. Protoc. 12, 2478–2492.

Kent, W.J., Zweig, A.S., Barber, G., Hinrichs, A.S., Karolchik, D. (2010). BigWig and BigBed: enabling browsing of large distributed datasets. Bioinformatics 26, 2204–2207.