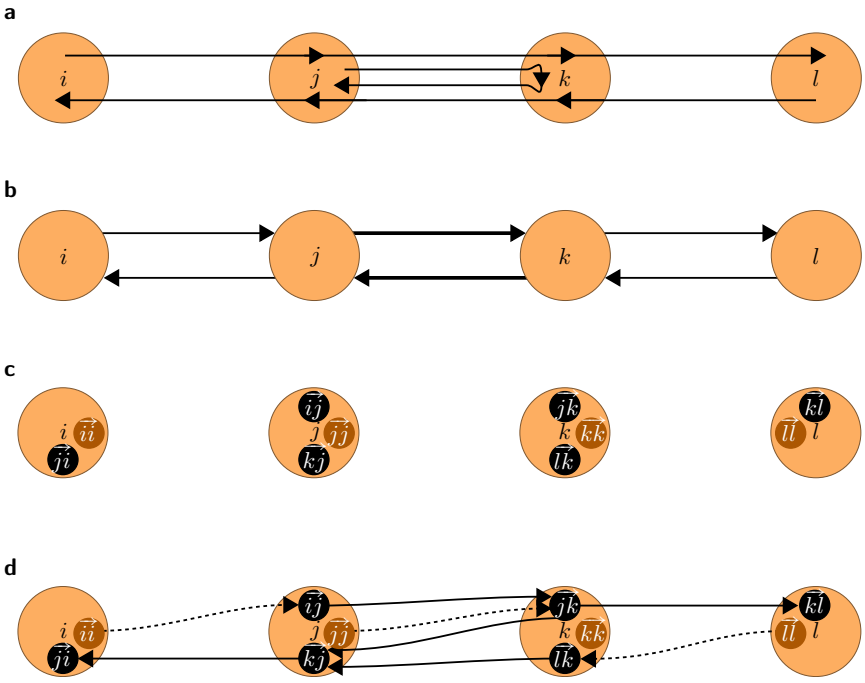
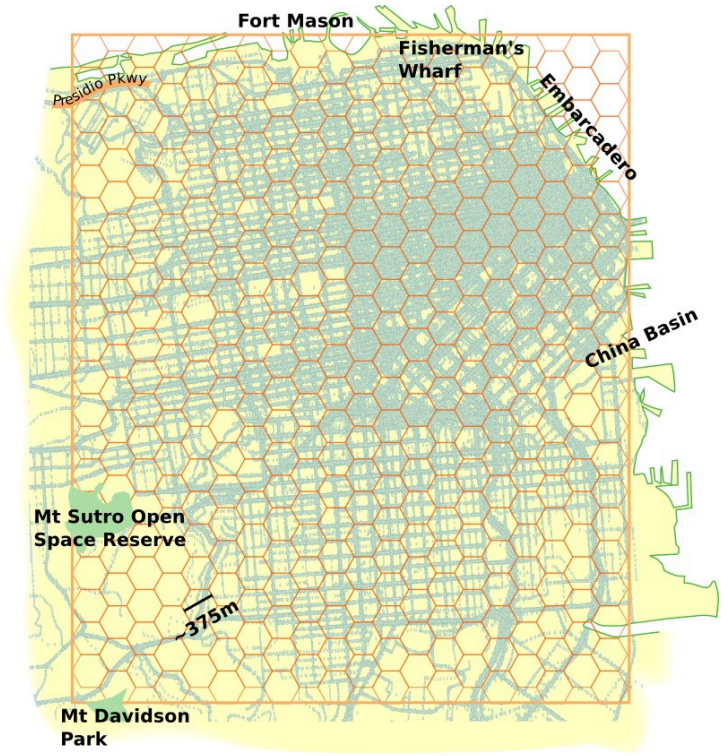
Supplementary Information

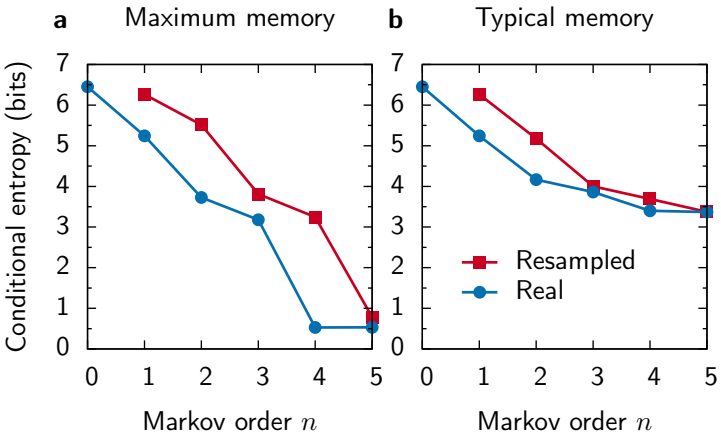
Supplementary Figures



Supplementary Figure 1 | Representing pathways with a standard network and a memory network. (**a**) The three pathways between four physical nodes in Supplementary Equation (1). (**b**) Standard network representation. (**c**) Memory network with memory nodes in black and self-memory nodes in brown. (**d**) Memory network representation. Dashed links in **d** from the self-memory nodes represent the first step of each pathway in **a**. We only use self-memory nodes in the cities dataset (see Supplementary Note 1). For the other networks, we only include the first step of each pathway in the weights of the teleportation scheme as described in the Methods section of the main text.



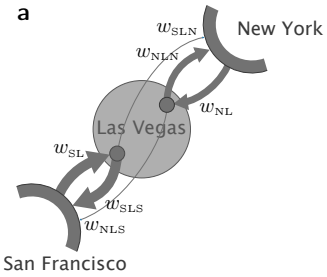
Supplementary Figure 2 | Taxi traffic in the San Francisco urban centre. We used the superimposed hexagonal grid to infer trigrams.



Supplementary Figure 3 | Predicting the next destination in air traffic. The entropy measures the uncertainty about a passengers next destination conditional on the sequence of already visited airports. The Markov order sets the maximum length of this memory. (**a**) Here we only consider passengers that have already visited at least n airports. (**b**) Here we consider all passengers and all their airport visits. Resampled variance is less than the size of the symbols.

Itinerary data

a



First-order Markov

b

$$w^{\mathrm{M1}}_{\mathrm{SLS}} = 32{,}260$$
$$w^{\mathrm{M1}}_{\mathrm{SLN}} = 22{,}162$$
$$w^{\mathrm{M1}}_{\mathrm{SL}} = 54{,}422$$
$$w^{\mathrm{M1}}_{\mathrm{NLN}} = 15{,}222$$
$$w^{\mathrm{M1}}_{\mathrm{NLS}} = 22{,}159$$
$$w^{\mathrm{M1}}_{\mathrm{NL}} = 37{,}381$$
$$w^{\mathrm{M1}}_{\mathrm{tot}} = 183{,}606$$

Second-order Markov

c

$$w^{\mathrm{M2}}_{\mathrm{SLS}} = 53{,}736$$
$$w^{\mathrm{M2}}_{\mathrm{SLN}} = 686$$
$$w^{\mathrm{M2}}_{\mathrm{SL}} = 54{,}422$$
$$w^{\mathrm{M2}}_{\mathrm{NLN}} = 36{,}698$$
$$w^{\mathrm{M2}}_{\mathrm{NLS}} = 683$$
$$w^{\mathrm{M2}}_{\mathrm{NL}} = 37{,}381$$
$$w^{\mathrm{M2}}_{\mathrm{tot}} = 183{,}606$$

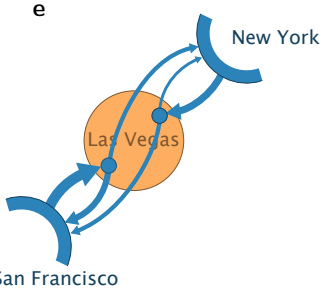
One-module description length

d

$$L(\mathsf{M}_1) = \frac{w_{\mathrm{tot}}}{w_{\mathrm{tot}}} H\Big(\overbrace{w_{\mathrm{SLS}} + w_{\mathrm{NLS}}}^{\substack{\text{Visit}\\ \text{San Francisco}}}, \overbrace{w_{\mathrm{SL}} + w_{\mathrm{NL}}}^{\substack{\text{Visit}\\ \text{Las Vegas}}}, \overbrace{w_{\mathrm{SLN}} + w_{\mathrm{NLN}}}^{\substack{\text{Visit}\\ \text{New York}}}\Big)$$

First-order Markov

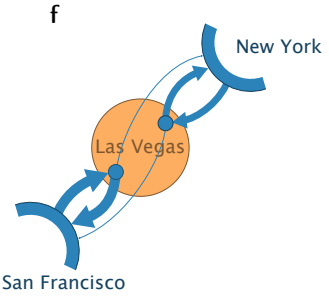
e



$$L(\mathsf{M}_1) = \mathbf{1.49}\,\text{bits}$$

Second-order Markov

f

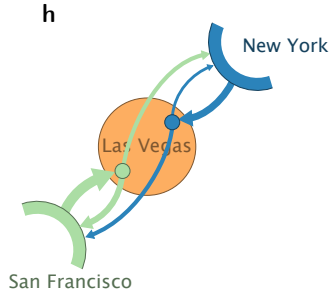


$$L(\mathsf{M}_1) = \mathbf{1.49}\,\text{bits}$$

Overlapping two-module description length

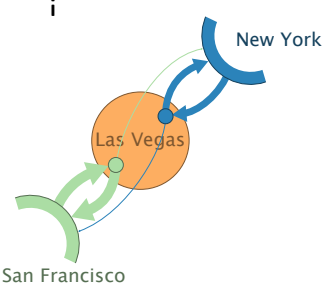
First-order Markov

h



Second-order Markov

i



g

$$L(\mathsf{M}_2) = \frac{w_{\mathrm{NLS}} + w_{\mathrm{SLN}}}{w_{\mathrm{tot}}} H\Big(\overbrace{w_{\mathrm{NLS}}}^{\substack{\text{Enter}\\ \text{green module}}}, \overbrace{w_{\mathrm{NLS}}}^{\substack{\text{Enter}\\ \text{blue module}}}\Big)$$

$$+ \frac{w_{\mathrm{SL}} + w_{\mathrm{NLS}} + w_{\mathrm{SLS}} + w_{\mathrm{SLN}}}{w_{\mathrm{tot}}} H\Big(\overbrace{w_{\mathrm{SL}}}^{\substack{\text{Visit}\\ \text{Las Vegas}}}, \overbrace{w_{\mathrm{SLS}} + w_{\mathrm{NLS}}}^{\substack{\text{Visit}\\ \text{San Francisco}}}, \overbrace{w_{\mathrm{SLN}}}^{\substack{\text{Exit}\\ \text{green module}}}\Big)$$

$$+ \frac{w_{\mathrm{NL}} + w_{\mathrm{NLN}} + w_{\mathrm{SLN}} + w_{\mathrm{NLS}}}{w_{\mathrm{tot}}} H\Big(\overbrace{w_{\mathrm{NL}}}^{\substack{\text{Visit}\\ \text{Las Vegas}}}, \overbrace{w_{\mathrm{SLN}} + w_{\mathrm{NLN}}}^{\substack{\text{Visit}\\ \text{New York}}}, \overbrace{w_{\mathrm{NLS}}}^{\substack{\text{Exit}\\ \text{blue module}}}\Big)$$

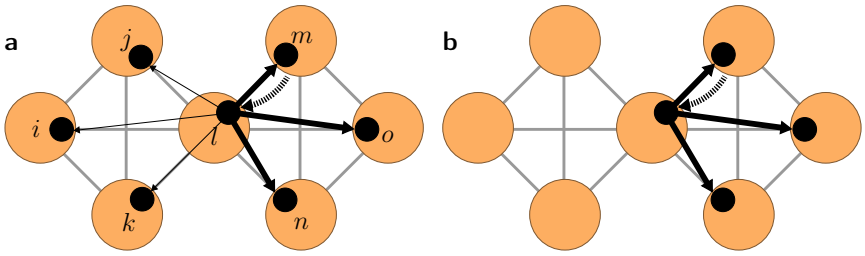
First-order Markov

$$L(\mathsf{M}_2) = \mathbf{2.12}\,\text{bits} = 0.24$$
$$+ 1.06$$
$$+ 0.82\,\text{bits}$$

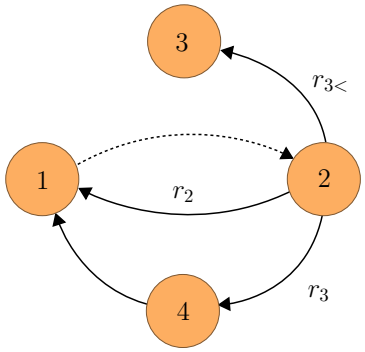
Second-order Markov

$$L(\mathsf{M}_2) = \mathbf{1.07}\,\text{bits} = 0.0075$$
$$+ 0.63$$
$$+ 0.44\,\text{bits}$$

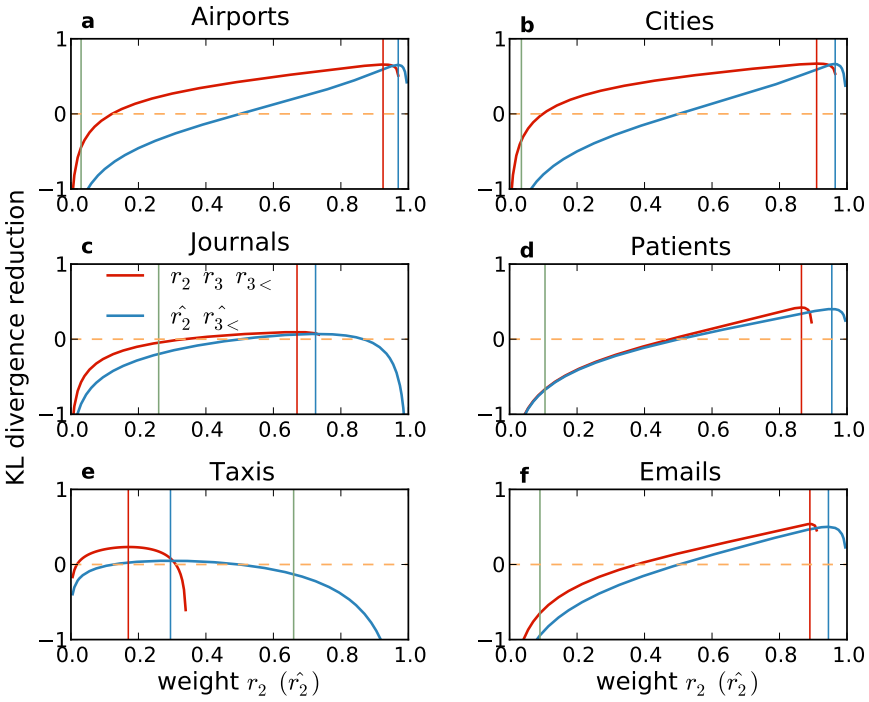
Supplementary Figure 4 | Second-order Markov dynamics reveal overlapping modules. Itineraries weighted by passenger number from the data presented in Supplementary Note 1, restricted to transfer traffic and trigrams that include San Francisco, Las Vegas, and New York. (**a**) Legend with labelled links between memory nodes. For example, w_{NLN} represents pathways that start in New York, continue to Las Vegas, and return to New York. (**b**) First-order Markov dynamics represented with links between memory nodes. (**c**) Second-order Markov dynamics represented with links between memory nodes. (**d**) The map equation for the one-module solution M_1 in **e**-**f**. Normalization of the weights in the entropy is implicit. (**e**) The description length of the one-module solution M_1 with memoryless flow. (**f**) The description length the a one-module solution M_1 with second-order Markov dynamics. (**g**) The map equation for the overlapping two-module solution M_2 in **h**-**i**. First line gives the average description length of movements between the two modules. Second line gives the average description length of movements within the green module. Third line gives the average description length of movements within the blue module. (**h**) The description length of the overlapping two-module solution M_2 with first-order Markov dynamics. (**i**) The description length of the overlapping two-module solution M_2 with second-order Markov dynamics.



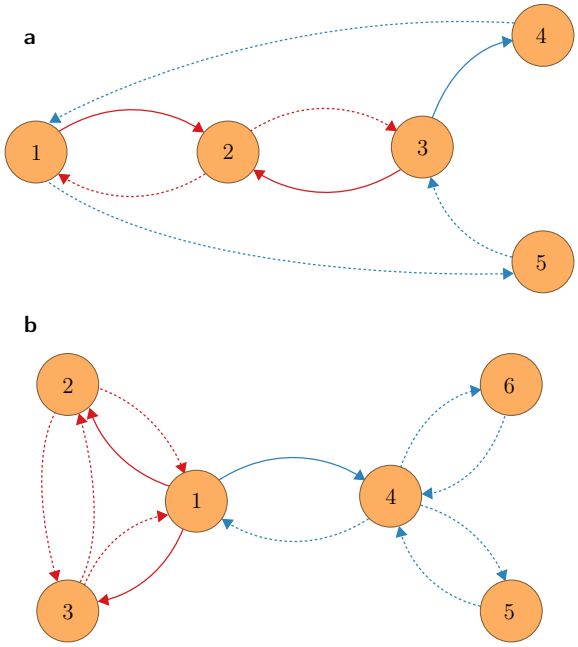
Supplementary Figure 5 | Interpreting link clustering and clique percolation as flow-based methods with second-order Markov dynamics. (**a**) Link clustering interpreted as a flow-based method with second-order Markov dynamics: The similarity between link $m \to l$ and other links connected to node l represented as similarity weighted out-links from memory node $\overrightarrow{ml}$ to connected memory nodes. (**b**) Clique percolation interpreted as a flow-based method with second-order Markov dynamics: Possible link sides of adjacent triangles sharing the link $m \to l$ represented as unweighted out-links from memory node $\overrightarrow{ml}$.



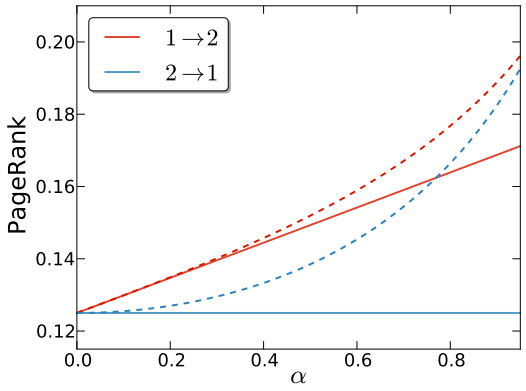
Supplementary Figure 6 | Illustration of the memory model. After performing a jump along the dashed link, a walker can either perform a return step, with probability r_2, a triangular step, with probability r_3, or an exploratory step, with probability $r_{3<}$.



Supplementary Figure 7 | Fitting the memory model. The plots show the KL divergence reduction in the memory model with all parameters r_2, r_3, $r_{3<}$ (red), and with restricted parameters $\hat{r_2}$, $\hat{r_{3<}}$ (blue) model. The vertical lines correspond to r_2 (red), r_3 (green) and $\hat{r_2}$ (blue).



Supplementary Figure 8 | Schematic networks without triangles and with reciprocated links. In **a**, there is no triangle and r_3 does not play a role. In **b**, all links are reciprocated and r_2 does not play a role. The stationary probability π increases/decreases with r_2 (**a**) or r_3 (**b**) for red and blue links respectively. For solid lines, this increase/decrease is predicted by the first-order perturbation in Supplementary Equation (17). For dashed lines, the probability of finding a walker on a memory node is uniform in the first-order approximation, and higher-order contributions are required to predict increase or decrease.



Supplementary Figure 9 | Linear approximation of memory effect on ranking. PageRank as a function of α for edges going from 1 to 2 and from 2 to 1, illustrated in Supplementary Fig. 8a, for $r_2 = 0.8$, $r_3 = 0.1$ and $r_{3<} = 0.1$. Solid lines correspond to the linear approximation given by Supplementary Equation (18) and Supplementary Equation (19).

Supplementary Tables

Supplementary Table 1 | Second-order Markov effects on constraints on flow

Network	Two-step return (%)		Three-step return (%)		Entropy rate (bits)	
	M1	M2	M1	M2	M1	M2
Airports	5.7 (5.7–5.7)	47 (47–47)	2.1 (2.1–2.1)	0.63 (0.63–0.64)	5.2 (5.2–5.2)	3.4 (3.4–3.4)
Cities	6.5 (6.5–6.5)	48 (48–48)	2.8 (2.8–2.8)	0.62 (0.62–0.62)	4.7 (4.6–4.7)	3.5 (3.5–3.5)
Journals	11 (11–11)	21 (21–21)	4.7 (4.7–4.7)	5.4 (5.4–5.5)	4.5 (4.5–4.5)	3.5 (3.5–3.5)
Patients	16 (16–18)	54 (51–55)	1.9 (1.7–2.1)	3.4 (2.0–3.2)	3.0 (2.5–2.6)	1.0 (0.92–1.0)
Taxis	20 (20–20)	10 (10–11)	6.8 (6.8–6.9)	10 (10–10)	2.2 (2.2–2.2)	1.1 (1.1–1.1)
Emails	14 (14–15)	58 (55–58)	5.2 (5.1–5.5)	2.7 (2.2–3.2)	3.0 (2.8–2.9)	1.3 (1.1–1.2)

M1 and M2 for results obtained with a first- and a second-order Markov model, respectively. Values in parentheses represent the 10th and 90th percentiles from the bootstrap analysis. All node averages are for physical nodes.

Supplementary Table 2 | Second-order Markov effects on community detection and ranking

Network	Module size (%)		Module assignments		Compression gain (%)	Ranking difference (%)
	M1	M2	M1	M2	M1→M2	M1→M2
Airports	93 (93–93)	5.1 (5.0–5.1)	1.2 (1.2–1.2)	6.2 (6.2–6.3)	13 (9.7–9.8)	8.2 (8.1–8.2)
Cities	32 (32–32)	5.3 (5.4–7.7)	1.8 (1.8–1.8)	3.7 (3.6–3.7)	4.7 (4.3–4.3)	3.7 (3.7–3.7)
Journals	14 (14–14)	15 (15–15)	1.8 (1.8–1.8)	3.4 (3.3–3.4)	4.7 (10–11)	9.7 (9.5–9.8)
Patients	7.3 (5.2–6.9)	1.9 (1.7–2.1)	5.0 (4.4–4.7)	4.7 (4.4–5.0)	30 (23–28)	22 (24–28)
Taxis	3.1 (2.9–3.3)	2.2 (2.2–2.4)	1.5 (1.5–1.6)	1.7 (1.7–1.8)	6.5 (6.5–7.2)	6.5 (6.5–7.1)
Emails	12 (11–13)	5.8 (4.7–6.3)	1.3 (1.4–1.6)	3.0 (2.5–2.7)	26 (23–27)	18 (20–24)

M1 and M2 for results obtained with first- and second-order Markov models, respectively. Values in parentheses represent the 10th and 90th percentiles from the bootstrap analysis. All node averages are for physical nodes.

Supplementary Table 3 | Memory reveals multidisciplinary journals in the scholarly literature

Field	PNAS		Science		Ecology		Plant Cell	
	M1	M2	M1	M2	M1	M2	M1	M2
Ecology	–	13	–	29	100	100	–	–
Cell biology	100	80	100	68	–	–	100	100
Mathematics	–	4.6	–	–	–	–	–	–
Statistics	–	1.5	–	–	–	–	–	–
Anthropology	–	–	–	1.6	–	–	–	–
Others	–	0.38*	–	1.4†	–	–	–	–

The relative assignment to each field in percentage. Number of other fields a journal is assigned to in parenthesis. *In 1 other field. †In 7 other fields.

Supplementary Table 4 | Fitted model parameters and relative reduction in KL divergence over a first-order Markov model

Network	$r_2(\%)$	$r_3(\%)$	$r_{3<}(\%)$	KL red. $(\%)$	$\hat{r_2}(\%)$	$\hat{\text{KL}}$ red. $(\%)$
Airports	93 (93-93)	2.8 (2.8-2.8)	4.5 (4.5-4.5)	66 (66-66)	97 (97-97)	65 (65-65)
Cities	91 (91-91)	3.2 (3.2-3.2)	5.6 (5.6-5.7)	67 (67-67)	96 (96-96)	66 (66-66)
Journals	67 (67-67)	25 (25-25)	7.6 (7.2-7.6)	9.6 (9.4-9.6)	72 (72-72)	7.0 (6.9-7.0)
Patients	86 (85-90)	10 (7.4-11)	3.4 (2.8-3.5)	40 (40-45)	95 (95-95)	40 (40-45)
Taxis	17 (16-17)	66 (65-66)	17 (17-18)	23 (22-23)	29 (29-30)	4.8 (4.8-5.0)
Emails	89 (88-89)	8.9 (8.4-9.8)	1.9 (1.8-2.2)	54 (51-55)	94 (94-95)	50 (47-51)

Values in parentheses are the 10th and 90th bootstrap percentiles. $\hat{\text{KL}}$ refers to the simplified model, where we only account for returning steps.

Supplementary Note 1
Data acquisition and processing

Here we describe how we estimate the transition probabilities from empirical data. We use empirical data that consist of large sets of multi-step pathways of varying length. To create memory networks that capture an nth-order Markov process, we count all pathways of length $n+1$ in the empirical pathways of length $n+1$ or longer. Take, as an example, the three pathways

$$i \to j \to k \to l,\ l \to k \to j \to i, \text{ and } j \to k \to j. \qquad (1)$$

In a standard network with physical nodes i, j, k, and l, we extract the four directed links $i \to j$, $j \to i$, $k \to l$, and $l \to k$ with weight 1 and the two directed links $j \to k$ and $k \to j$ with weight 2, as shown in Supplementary Fig. 1a. These links are all pathways of length 1 that can be extracted from the three pathways in Supplementary Equation (1). The weight $W(i \to j)$ of a link corresponds to the number of times the link occurs in the pathways. With $W(i \to j) = 0$ if there is no link from i to j, the transition probabilities take the form

$$p(i \to j) = \frac{W(i \to j)}{\sum_k W(i \to k)}. \qquad (2)$$

Instead, in a memory network with memory nodes $\overrightarrow{ij}$, $\overrightarrow{jk}$, $\overrightarrow{kl}$, $\overrightarrow{lk}$, $\overrightarrow{kj}$, and $\overrightarrow{ji}$, we extract the five directed links $\overrightarrow{ij} \to \overrightarrow{jk}$, $\overrightarrow{jk} \to \overrightarrow{kl}$, $\overrightarrow{lk} \to \overrightarrow{kj}$, $\overrightarrow{kj} \to \overrightarrow{ji}$, and $\overrightarrow{jk} \to \overrightarrow{kj}$, all of weight 1 in this case, as shown in Supplementary Fig. 1d. These links are all pathways of length 2 that can be extracted from the three pathways in Supplementary Equation (1). With $W(\overrightarrow{ij} \to \overrightarrow{jk})$ to denote the weight of a link $\overrightarrow{ij} \to \overrightarrow{jk}$ and $W(\overrightarrow{ij} \to \overrightarrow{jk}) = 0$ if there is no link from $\overrightarrow{ij}$ to $\overrightarrow{jk}$, the transition probabilities take the form

$$p(\overrightarrow{ij} \to \overrightarrow{jk}) = \frac{W(\overrightarrow{ij} \to \overrightarrow{jk})}{\sum_l W(\overrightarrow{ij} \to \overrightarrow{jl})}. \qquad (3)$$

This formalism can be extended to memory networks that capture even higher-order Markov processes, but here we focus on describing memory networks that capture second-order Markov processes. In this case, links between physical nodes in the standard network representation take the role of memory nodes in the memory network representation. Therefore, a memory network is a form of line graph, but we use the term memory network to highlight our purpose with this representation: to capture movements between physical nodes with transition rates that depend on the past. In network science, line graphs have recently been introduced for a different purpose, namely, to move the focus from nodes to links as a computational trick to detect overlapping modules in networks[1,2]. Supplementary Figure 1 shows that the memory network contains more information than a line graph derived from the standard network would. Memory node $\overrightarrow{ij}$, for example, corresponds to the link $i \to j$ in the standard network representation. However, this link is not connected with both link $j \to k$ and link $j \to i$, as the standard network suggests, but only with link $j \to k$, as the memory network shows.

With very long empirical pathways that generate ergodic processes on the memory networks, we could directly use (7) in the main manuscript without any extra work. In practice, however, most pathways are between three and six steps long and boundary effects can influence the analysis. Also, as with directed standard networks, a Markov process on a memory network is rarely ergodic without introducing a small teleportation probability as described in the Methods section of the main manuscript.

Since we are interested in second-order Markov dynamics, it is convenient to store all pathway data as trigrams. From the pathways in Supplementary Equation (1), we store the trigrams in the following way:

```
i i j 1
i j k 1
j k l 1
l l k 1
l k j 1
k j i 1
j j k 1
j k j 1
```

The last number in each line gives the frequency of the corresponding trigram. It is straightforward to calculate the transition probability (6) in the main manuscript between memory nodes from the list of trigrams. For instance, $p(\overrightarrow{ij} \to \overrightarrow{jk}) = 1$, $p(\overrightarrow{jk} \to \overrightarrow{kl}) = 1/2$, etc.

To be able to initiate dynamics in physical nodes, we repeat the first physical node of pathways twice. We call the corresponding memory nodes self-memory nodes. They represent flow in a physical node that was in the same physical node in the previous step. For example, the first trigram of the pathway $i \to j \to k \to l$ is `i i j 1` with added self-memory node $\overrightarrow{ii}$ (the three trigrams with self-memory nodes above correspond to the dashed links in Supplementary Fig. 1d). In this way, the second pair of nodes in each line forms the link between physical nodes in a standard network representation and it becomes straightforward to obtain M1 data. The procedure is to simply ignore the first node in the trigrams and, for each link between two physical nodes i and j, aggregate all weights to $W(i \to j)$. We stress that we use the self-memory nodes only to obtain link weights and M1 data for teleportation steps and not for regular steps (the city network is an exception; see Supplementary Note 1).

When parsing the data, the first line of the trigram data reads literally: "*There is one link between self-memory node $\overrightarrow{ii}$ and memory node $\overrightarrow{ij}$.*" The meaning is: "*One pathway starts in i and continues to j*," and we increase the weight of memory node $\overrightarrow{ij}$ by 1. The second line of the trigram data reads literally: "*There is one link between memory node $\overrightarrow{ij}$ and memory node $\overrightarrow{jk}$.*" The meaning is: "*One pathway in j came from i and continues to k*," and we add a link from memory node $\overrightarrow{ij}$ to memory node $\overrightarrow{jk}$ and increase the weight of memory node $\overrightarrow{jk}$ by 1.

Cities and Airports

We compiled the airline pathways from the Airline Origin and Destination Survey (DB1B) made public by the Research and Innovative Technology Administration (RITA). The data contain each stop on 19,415,369 itineraries between 464 airports in the US with average pathlength 3.3 (21% of length two, 53% of length three, 5% of length four, 19% of length five, 1.4% of length six, and 1.0% of length seven or longer). Note that the origin is included in the path length, such that a path length of three corresponds to two flight legs. We used data from the first three quarters of 2011.

In the city memory network, we aggregated all airports within a radius of 50 kilometres. Since the airport data have clear starts and stops — a passenger is based in a city and most often returns

to the same city at the end of the itinerary — we also include the home city itself in the analysis. In this way, we can better capture real passenger traffic and extend the analysis beyond transfer traffic. We thus represent the home city with the corresponding self-memory node in the memory network and, unlike in the other memory networks, include the self-memory nodes in the analysis. For example, we represent a pathway $i \dots j \dots k \dots j \dots i$ going from city i to city k with transfer in city j and back with the trigrams

```
i i j 1
i j k 1
j k j 1
k j i 1
j i i 1
```

and include the first and last trigrams in the analysis.

In the airport memory network, we focused on transfer traffic and did not include self-memory nodes. For example, a pathway $i \dots j \dots k \dots j \dots i$, representing an itinerary from airport i to airport k with transfer at airport j and back, is represented by the trigrams

```
i i j 1
i j k 1
j k j 1
k j i 1
```

with the first trigram `i i j 1` included only for calculating the teleportation weight in the PageRank analysis.

Journals

The journal citation data were extracted from JSTOR (www.jstor.org), a not-for-profit digital library that includes 2,227 journals and 8,227,537 citations among these journals. In order to capture memory, we extracted the underlying article-level citations. This included 1,787,351 unique articles that cite at least one other JSTOR article or received a citation from another JSTOR article.

JSTOR does not represent the full universe of scholarly content. For example, the journal *Nature* is not included in this subset. In addition, physics, engineering, and computer science are not well represented in this corpus of articles. However, it did offer several advantages. First, JSTOR made its data available for research. Second, the JSTOR corpus has both article- and journal-level data, which were necessary for building memory networks.

For each article A in JSTOR, we searched all articles A_{out} cited by A and all articles A_{in} citing A. If we found at least one cited and one citing article, then, for each cited article, we picked a random citing article and formed the trigram

$$A_{\text{in}} \to A \to A_{\text{out}}, \tag{4}$$

which we mapped to the trigram between the corresponding journals

$$J_{\text{in}} \to J \to J_{\text{out}}. \tag{5}$$

Finally, we aggregated all journal trigrams into the weighted memory network. By sampling memory networks many times, we found that choosing a random citing article did not affect our ranking or community detection results[3] (see Supplementary Note 2).

Patients

The patient data derive from a database of inpatient care at hospitals in Stockholm, Sweden, during 2001 and 2002[4]. The full dataset consists of 295,108 individuals who entered at least one of 702 wards at hospitals or nursing homes in 52 different locations. Our anonymised data were compiled by Fredrik Liljeros and are a sample of 365 days from the original data. Since we are only interested in patients who entered three or more wards, the data contain records from fewer individuals than the original data and are limited to patient movements between 402 wards.

Taxis

We included the taxi data because we were interested in analysing a real-world system lacking strong return flow. With this data, we can contrast the dynamics in the other networks. The data are from GPS receivers in smartphones of taxi drivers from the Uber taxi company in the San Francisco region[5]. We further processed the data in the following way. First, because most of the taxi traffic is in the metropolitan area of San Francisco, we limited our analysis to the rectangle limited by longitudes 122.456 122.388 West and latitudes 37.748 37.808 North. In that rectangle, we superimposed a hexagonal grid of 20 x 20 hexagons, depicted in Supplementary Fig. 2. The distance between opposite sides in each hexagon of the grid is approximately 375 meters, or about the length of two city blocks. Indeed, we observed some shorter trips in the dataset, but not many. For each of 25,000 taxi trajectories in the data set, we built chains of 2-d integer coordinates corresponding to the hexagon where the taxi is driving at a given moment. That is, geographical hexagons are the nodes of the network, and there is a link between two hexagons if they share an edge. Finally, we took triplets of consecutive hexagons visited by a taxi and summarized them in trigrams, as described above. For calculating the weight of each trigram, we used the number of times that a taxi trajectory contained the trigram.

To validate that our results are not affected by the particular choice of grid structure, we also derived memory networks from 10 x 10 and 40 x 40 hexagonal grids. The results were qualitatively the same, with small quantitative differences.

Emails

The Enron email dataset[6] comprises emails between 146 users disclosed during the trial following the Enron scandal. We aggregated all email addresses for each single user, and used a total of 116,525 messages between users as links. In order to establish succession between messages, we looked to common word-triplets present in the subject, overlap between the *to* and *from* fields, and message date-time. Because we linked messages using their common word-triplets, many messages were linked to many other messages through their subject line headings. We removed the redundancy by leaving only one incoming link for each message, namely, the one with the latest possible time-stamp. Finally, we broke the pathways into trigrams as described above.

Supplementary Note 2
Significance analysis with resampling

To verify that our results are based on sufficient data, we performed bootstrap resampling of pathways for all summary statistics reported in the main manuscript and below, and surrogate data testing of the entropy rate to estimate the Markov order.

Bootstrap resampling

For each dataset, we generated 100 bootstrap replicas by resampling the pathways with replacement and then constructed the networks from the replicas. Each dataset contains a large number of pathways distributed over a smaller number of unique pathways, such that each unique pathway has a weight given by the number of that unique pathway in the dataset. Therefore, we generated a single bootstrap memory network by first resampling the weights of all unique pathways from a multinomial distribution and then breaking the long pathways into trigrams (for patients, taxis, and emails, we only had access to trigrams and directly resampled their weights). For a given dataset, we used a multinomial distribution with as many categories as unique pathways in the dataset and with probabilities for the categories proportional to the weights of the unique pathways. From this multinomial distribution, we performed as many trials as the total number of pathways in the dataset and aggregated the outcome to resampled weights for all unique pathways.

For the journal network, we constructed the bootstrap replicas differently, because the journal memory network is not constructed from pathways but from chaining article citations. The procedure described in Supplementary Note 1 above involves a random step, and we simply generate the bootstrap replicas by repeating this procedure. Note that this procedure does not generate any variation in the standard network, but it is anyway the significance of the second-order Markov results that we are interested in.

For each set of summary statistics, we calculated the bootstrap confidence interval by ordering 100 bootstrap estimates and eliminated the ten smallest and ten largest estimates. In this way, the remaining estimates span the 90% bootstrap percentile confidence interval. In general, we report the lower and upper limits of this interval. If there are many trigrams with only a few observations in the data, a summary statistic of the raw data can lie outside of the bootstrap confidence interval. Nevertheless, there can still be a memory effect. For a significant memory effect, what really matters is non-overlapping confidence intervals for M1 and M2 dynamics.

Surrogate data testing

For the memory effects of individual nodes reported in Fig. 2 of the main manuscript, we also performed hypothesis testing to verify that our results are based on sufficient data. Our null hypothesis was that the flow is a first-order Markov process, and we used the conditional entropy at each node as a test statistic. Assuming that the null hypothesis is true, we estimated the probability that the conditional entropy of the second-order Markov process is at least as low as the observed value. We estimated this probability, the p-value, with bootstrap resampling[7] and rejected the null hypothesis if the p-value was lower than 0.10.

To resample the data at each node i, we used the so called symbolic surrogate procedure with constrained probabilities surrogates[8]. We first collected all trigrams $x_1x_2x_3$ that pass through node i in the first step ($x_2 = i$). These trigrams form a set of n pairs

$$\left\{(x_1^1,x_3^1),(x_1^2,x_3^2),\ldots,(x_1^n,x_3^n)\right\}, \tag{6}$$

with all steps before and after node i. To resample these pairs, we randomized the order of the set of nodes visited before node i, $\left\{x_1^1,x_1^2,\ldots,x_1^n\right\}$ and created random pairings with the set of nodes visited after i, $\left\{x_3^1,x_3^2,\ldots,x_3^n\right\}$, thereby destroying any memory effect. After aggregating the transition probabilities for the random pairings, we calculated the conditional entropy for the randomized data. We repeated this procedure, random resampling and calculation of conditional entropy, as many times as needed to conclude that the p-value is lower or higher than 0.10. We used the Clopper-Pearson method[9] with a 90% confidence interval of the p-value to determine the stop condition.

For the air traffic data, we have sufficiently many long pathways to perform this analysis also for higher-order Markov models. For order n, we extracted all $n+1$-grams from the data. Since we use the conditional entropy as a test statistics, we estimate the average amount of information necessary to determine the destination of a passenger, given information about the sequence of airports the passenger has visited. In this way, the Markov order sets the horizon of how much information an observer at an airport has about passengers to determine their next step. For each Markov order n, we performed two tests: one in which we only included n-grams of length $n+1$, and one in which we also included all shorter n-grams of length 2, 3, $\ldots n$ from each pathway. Excluding shorter n-grams corresponds to consider only passengers that already have visited at least n airports, which we refer to as the *maximum memory* of passengers at airports. Including shorter n-grams corresponds to consider all passengers and all their airport visits, which we refer to as *typical memory* of passengers at airports.

For each Markov order n, we performed the same resampling procedure as described above. We split all $n+1$-grams into two sets, one that consists of the n first airports of each $n+1$-gram and one that consists of the n last airports of each $n+1$-gram. Then we generated resampled $n+1$-grams by randomly recombining the two sets such that each resampled $n+1$-gram begins with an n-gram from the first set and ends with an n-gram from the other set, with $n-1$ airports overlapping in the middle. In this way, the resampled $n+1$-grams will be of Markov order $n-1$. In the typical memory approach, we only resampled the longest $n+1$-grams, since all other 2-, 3-, $\ldots n$-grams are of order $n-1$ or lower. We repeated this resampling 100 times for each Markov order n and compared the actual conditional entropies with the ones given by the null hypothesis that they are generated from Markov order $n-1$. Unlike in the main text where we weight the conditional entropies by PageRank, here we use the actual visit frequencies.

Supplementary Figure 3 shows the results. Even if air traffic has statistically significant memory effects up to Markov order four (Supplementary Fig. 3a), shorter itineraries dominate (73% of all itineraries are of length three or shorter) and a second-order Markov model accurately captures the typical memory dynamics (Supplementary Fig. 3b). For typical memory, the conditional entropy drops by 1.1 bits from first to second Markov order but only by 0.3 bits from second to third Markov order. Both results are statistically significant and we conclude that a second-order Markov model seems to successfully balance model complexity and accuracy. A more thorough model selection analysis is beyond the scope of this work.

Finally, in Supplementary Table 1 we report the conditional entropies of first- and second-order Markov dynamics together with the two-step and three-step return rates. Here we also complement the results reported in Table 1 of the main manuscript with the 10th and 90th percentiles of the bootstrap values. The bootstrapping shows that the memory effects are significant. For air-

ports and cities, the data are sufficiently rich that more than two digits are significant. For the patient data, however, many pathways through the hospitals are used by only one or a few patients. Therefore, the bootstrap estimates vary more and sometimes exclude the summary statistics of the raw data. Nevertheless, there is a significant effect of second-order Markov dynamics in all cases.

Supplementary Note 3
Community detection of memory networks

We have seen that a second-order Markov model has important effects on dynamic processes on networks. To better understand these effects, we simplified the dynamics and highlighted the important structures of the dynamics with community detection, currently the best way to comprehend dynamics on a large scale[10]. With community detection, we can compare the structure of first- and second-order Markov dynamic. Since we are interested in the dynamics, we have chosen to work with the flow-based map equation framework[11]. Alternative flow-based methods exist[12,13], but the map equation framework easily allows us to maintain the mechanics of the method and only modify the dynamics. That is, we can cluster physical nodes and use memory nodes for controlling the dynamics. This advantageous feature allows us to efficiently compare the first- and second-order Markov dynamics. Since we are interested in overlapping modules, we build our new method on a generalization of the map equation to overlapping modules[14].

The map equation

The map equation framework is an information-theoretic approach that takes advantage of the duality between compressing data and finding regularities in the data. Given module assignments M of all nodes in the network, the map equation measures the description length $L(\mathsf{M})$ of a random walker who moves within and between modules from node to node by following the links between the nodes[15]:

$$L(\mathsf{M}) = q_{\curvearrowright} H(\mathscr{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathscr{P}^i) \tag{7}$$

Here the entropy $H(\mathscr{Q})$ measures the average per-step description length of movements between modules derived from module-enter frequencies $\mathscr{Q}$ of all m modules and $H(\mathscr{P}^i)$ measures the average per-step description length of movements within module i derived from node-visit and module-exit frequencies $\mathscr{P}^i$. The description lengths are weighted by their frequency of use, $q_{i\curvearrowright}$ and $p^i_{\circlearrowright}$, respectively. The visit frequencies can be obtained by first calculating the PageRank of nodes and links with smart teleportation as described in the Methods section of the main text, or directly from the data if the links represent flow themselves.

In any case, finding the optimal partition of the network by assigning each node to one or more modules corresponds to testing different node assignments and picking the one that minimizes the map equation.

The challenge is to handle the large search space of possible solutions when nodes can be assigned to any number of overlapping modules. Therefore, we limit the search space here and only allow each memory node to be assigned to a single module. In this way, the efficient[16] search algorithm for hard partitions in *Infomap*[17,18] can be used with only small modifications. Instead of applying the search algorithm on the standard network, we apply it on the memory network that contains transition information between memory nodes (links). The method is thus a form of link partitioning[1,2]. The search algorithm initiates each memory node in its own module and proceeds as *Infomap* for hard partitions of regular nodes[17], with one important difference: When two or more memory nodes of the same physical node are assigned to the same module, the description length must capture the fact that the memory nodes share the same codeword. That is, to obtain the visit frequency of a physical node in a module, we sum the visit frequencies of all memory nodes of that physical node in the module. We then use this visit frequency to derive the optimal codeword length. This procedure is essential to ensure that the map equation measures the optimal description length of a random walker navigating between physical nodes. In this way, the compression algorithm remains the same and only the dynamics change. Moreover, we ensure that the community detection results only depend on memory effects by representing first-order Markov dynamics flow in a memory network, with each memory node having the out-links of its corresponding physical node in the standard network.

Supplementary Figure 4 illustrates the mechanics of the map equation for first- and second-order Markov dynamics. The first-order passenger trigrams in b are derived from the actual trigrams in c in two steps. We first derived the normalized out-links of the two memory nodes in Las Vegas. Since these memory nodes should represent first-order Markov dynamics, their out-links are identical and equal to the proportion of passengers flying to San Francisco and New York, respectively. We then multiplied these transition probabilities by the number of passengers that arrive in Las Vegas from San Francisco and New York, respectively. In this way, for example,

$$w^{\mathrm{M1}}_{\mathrm{SLS}} = w^{\mathrm{M2}}_{\mathrm{SL}} \frac{w^{\mathrm{M2}}_{\mathrm{SLS}} + w^{\mathrm{M2}}_{\mathrm{NLS}}}{w^{\mathrm{M2}}_{\mathrm{SL}} + w^{\mathrm{M2}}_{\mathrm{NL}}}, \tag{8}$$

where

$$w^{\mathrm{M2}}_{\mathrm{SL}} = w^{\mathrm{M2}}_{\mathrm{SLS}} + w^{\mathrm{M2}}_{\mathrm{SLN}} \tag{9}$$

and

$$w^{\mathrm{M2}}_{\mathrm{NL}} = w^{\mathrm{M2}}_{\mathrm{NLN}} + w^{\mathrm{M2}}_{\mathrm{NLS}}. \tag{10}$$

Further,

$$w^{\mathrm{M1}}_{\mathrm{tot}} = w^{\mathrm{M2}}_{\mathrm{tot}} = 2w^{\mathrm{M2}}_{\mathrm{SL}} + 2w^{\mathrm{M2}}_{\mathrm{NL}} \tag{11}$$

corresponds to the total passenger weight involved in the set of two consecutive flight legs illustrated in Supplementary Fig. 4.

Memory and heuristic algorithms

The link clustering[2] and clique percolation[19] methods can be seen as trying to account for second-order Markov dynamics. They operate by increasing connectivity within modules and decreasing connectivity between modules, albeit in different ways. Below we establish this flow interpretation of the two methods.

The link clustering method measures the similarity between two links $k \to i$ and $k \to j$ connected at a common "keystone" node k as the ratio between all shared nodes and the total number of nodes reached in two steps from the keystone node via the links. That is, the similarity is $|n_+(i) \cap n_+(j)|/|n_+(i) \cup n_+(j)|$, where $n_+(i)$ is the set of neighbours of i. Supplementary Figure 5a illustrates the similarities between nodes. With the keystone node in the middle, the dashed link is connected to five

other links. The similarity is 1 with the links in the same module, because they share all nodes two steps from the keystone node via the links. However, the similarity is only 1/7 with the links in the other module, because they only share the keystone node of all seven nodes that can be reached in two steps from the keystone node via the links. With these link weights between links, it is clear that the link clustering method identifies two modules overlapping at the node in the centre. How can we interpret this machinery as constraints on flow? With similarities represented as links between memory nodes, as in Supplementary Fig. 5a, we see that a random walker would rarely switch between the modules. Conditional on being at the node in the centre, and assuming that the link self-similarity is 1, the transition rate decreases from 1/2 to 1/7 with weights derived according to the link community procedure. In this way, the persistence time increases in the two modules and allow for efficient compression of the flow.

Clique percolation identifies a module as the maximum set of nodes that can participate in a percolation of adjacent cliques. A clique is a fully connected sub-graph and two cliques are adjacent if they share all nodes but one. Here we consider sub-graphs of size three, triangles, such that two triangles are adjacent if they share two nodes or, equivalently, one side. Supplementary Figure 5b illustrates. Since a triangle can percolate between the leftmost four nodes or the rightmost nodes, the method identifies two modules overlapping at the node in the centre. How can we interpret this process as constraints on flow? Assume that a random walker steps from node m to node l. The two adjacent triangles that share this link $m \to l$ have link sides $l \to o$ and $l \to n$, respectively, connected to node l. We now restrict the random walker to only move along those links, or back from where it came. If we use memory nodes to represent these constraints, as in Supplementary Fig. 5b, the transition rate between the two modules drops to zero. In fact, the random walker will be blocked by the very same module boundaries as given by the clique percolation method. Again, the persistence time increases in the two modules and allow for efficient compression of the flow.

By interpreting the machinery of the two methods as constraints on flow, we see that they can be seen as trying to infer second-order Markov dynamics from the structure of the standard network. Moreover, those constraints give longer persistence time in modules. And indeed, as we have seen from using real data of second-order dynamics, the persistence time in modules does increase when accounting for higher-order memory effects. As a result, this flow interpretation establishes an interesting connection between two heuristic methods that operate on standard networks and our inherently flow-based method that operates on memory networks. We conclude that this flow interpretation provides more principled grounds for link clustering and clique percolation.

Results of the statistical analysis

We summarize with the results of the community detection analysis and the closely related ranking in Supplementary Table 2. In addition to the results already presented in Table 1 of the main manuscript, here we also report the 10th and 90th percentiles from the bootstrap analysis. Because of the greedy and stochastic nature of the search algorithm, the confidence intervals are wider than for the entropies and return rates reported in Supplementary Table 1. Nevertheless, there is a significant difference between the structure of first- and second-order Markov dynamics. In a second-order Markov model, the dynamics are confined in smaller and more overlapping modules.

Supplementary Note 4
Modelling second-order Markov effects

So far, we have used empirical data and studied the effects of second-order Markov dynamics on community detection, ranking and spreading processes. Here we outline a different line of research aimed at identifying simple mechanisms for explaining the effects of memory in networks. The modelling approach can be seen as an initial step in bringing together complementary knowledge from previously disconnected areas of research: Biologists have used empirical data to model animal movements in 2D with correlated random walks[20–22], computer scientists have used web logs to predict web surfer behaviour with higher-order Markov models[23,24], and physicists have used theoretical models to study dynamics on networks with biased random walks[25–27]. By combining the empirical work for prediction with the theoretical work for mechanistic understanding, we are in a good position to better understand the effects of memory in integrated systems.

To combine the approaches, we developed a simple network memory model that, fitted to data, can capture some basic features of second-order Markov dynamics in real systems. In addition to its explanatory power, the memory model also summarizes the dynamics of a system and makes it easy to compare the dynamics between different systems. Below, we first describe the memory model, then we show a procedure for fitting the model parameters to real data, and finally we illustrate, with ranking as an example, how this modelling approach can be used for a mechanistic understanding of the effects of second-order Markov dynamics.

The memory model

We build a tractable model for memory dynamics by coarse-graining the description of second-order Markov data. We define three different types of transition between nodes: a *return step* r_2, where the walker goes from i to j to i; a *triangular step* r_3, where the walker goes from i to j to a neighbour of i; and an *exploratory step* $r_{3<}$, for which the destination of the step is neither of those previously described. These events correspond to transitions of the types $\overrightarrow{ij} \to \overrightarrow{ji}$, $\overrightarrow{ij} \to \overrightarrow{j\sigma(i)}$, and $\overrightarrow{ij} \to \overrightarrow{j\gamma(i)}$, where $\sigma(i)$ is a member of the set of neighbours of i and $\gamma(i)$ is a member of the set of nodes different from i and the set of neighbours of i (in principle, we can chose the set of either in-neighbours, out-neighbours, or neighbours in an undirected sense. In this discussion, we chose the latter option for the sake of simplicity).

The memory model is defined by assigning a different prevalence $w(\overrightarrow{ij} \to \overrightarrow{jx})$ to each type of transition, r_2, r_3, and $r_{3<}$, respectively (see Supplementary Fig. 6), where the index underlines the length of the cycle associated with the process. For instance, the probability of performing a return step from $\overrightarrow{ij}$, and thus of observing a cycle of length 2, is

$$p_{\text{return}} = r_2/(r_2 + r_3|\sigma(i)| + r_{3<}|\gamma(i)|) \tag{12}$$

if $\overrightarrow{ji}$ exists; otherwise, it is zero. $|\sigma(i)|$ is the number of elements in $\sigma(i)$. Tuning the values of r_x gives more or less importance to each type of step. Importantly, the values of the parameters can easily be evaluated in empirical data, as shown below. Without loss of generality, we impose the constraint $r_2 + r_3 + r_{3<} = 1$, such that r_x can be understood as the probability of an event of type x occurring. The memory model is described by transition probabilities $\hat{p}(\overrightarrow{ij} \to \overrightarrow{jk})$, approximating the original transition probabilities $p(\overrightarrow{ij} \to \overrightarrow{jk})$, and depending on the value of the above parameters and on the type of transitions between $\overrightarrow{ij}$ and $\overrightarrow{jk}$.

Fitting the memory model

We now seek to find parameter values of r_2, r_3, and $r_{3<}$ to model dynamics as close as possible to observed M2 data. The model is thus fitted by minimizing the difference between the observed transition probabilities $p(\overrightarrow{ij} \to \overrightarrow{jk})$ measured from the trigrams and the transition matrix of the model $\hat{p}(\overrightarrow{ij} \to \overrightarrow{jk})$. To do so, we look for the values of r_2 and r_3, minimizing the Kullback−Leibler (KL) divergence

$$D_{\mathrm{KL}} = \sum_{\overrightarrow{ij}} \pi(\overrightarrow{ij}) \sum_{\overrightarrow{jk}} p(\overrightarrow{ij} \to \overrightarrow{jk}) \log \frac{p(\overrightarrow{ij} \to \overrightarrow{jk})}{\hat{p}(\overrightarrow{ij} \to \overrightarrow{jk})}, \qquad (13)$$

where $\pi(\overrightarrow{ij})$ is, as before, the PageRank of node $\overrightarrow{ij}$. Minimizing the KL divergence is equivalent to maximizing the log-likelihood[28], but since we use the conditional entropy to quantify the constraints on flow, we find it natural to use the information-theoretic KL divergence as the objective function.

In practice, in order to minimize the KL divergence, we first analytically derive its partial derivatives with respect to r_2 and r_3 ($r_{3<}$ does not appear in the expression of the KL divergence because of the normalization $r_{3<} = 1 - r_2 - r_3$). Then we set the derivatives to zero and iteratively solve the two coupled equations with a simple bisection method.

The parameters found by performing this optimization are summarized in Supplementary Table 4. The table also reports values for $\hat{r_2}$, the best parameter in a simplified model where we impose that $r_3 = r_{3<}$, and thus only differentiate between return steps and other type of steps. We quantified the relative KL reduction as 1 minus the ratio between the optimized KL divergence and the unbiased first-order Markov model with $r_2 + r_3 + r_{3<} = 1/3$ ($\hat{\mathrm{KL}}$ refers to the simplified model); high values of the KL reduction imply a higher relative gain of fitting the data with the memory model over a first-order Markov model. In most cases, we observe that the inclusion of memory through the parameters r_2, r_3, and $r_{3<}$ significantly improves the accuracy of the modelling. A majority of networks show a high value of r_2, as expected. We also observe that the reduction induced by relaxing the constraint $r_3 = r_{3<}$ tends to be small, suggesting that the definition of return steps is the most important ingredient in producing realistic pathways. Supplementary Figure 7 shows the KL divergence reduction when tuning r_2 and $\hat{r_2}$.

It is worth mentioning that the optimization procedure provides a unique solution because the search landscape is indeed very smooth. For example, Supplementary Fig. 7 shows that the KL divergence has a unique minimum (the reduction has a maximum) as a function of r_2 with r_3 fixed. Also the reversed scenario, fixed r_2 and variation in r_3, has a similar smooth form with a unique extremum. Moreover, the bootstrap analysis shows that the optimal parameters are very robust.

Analytical analysis of second-order Markov effects on ranking

Here we use the model to illustrate the effects of second-order Markov dynamics on ranking. We use schematic networks and, for simplicity, we focus on unweighted networks. In this case, $W(i \to j)$ is simply the adjacency matrix A_{ij} of the network, with $A_{ij} = 1$ if there is a link going from i to j and zero otherwise. It is also useful to introduce the in- and out-degrees of each node defined by $\sigma_j^{\mathrm{in}} = \sum_i A_{ij}$ and $\sigma_i^{\mathrm{out}} = \sum_j A_{ij}$. As a first step, we focus on the basic case when each type of transition is equally probable, $r_2 + r_3 + r_{3<} = 1/3$. In this case, the memory model is equivalent to a standard Markov random walk on the physical network. To show this, we first note that (7) in the main manuscript reduces to

$$P(\overrightarrow{jk};t+1) = \sum_i P(\overrightarrow{ij};t) \frac{A_{jk}}{\sigma_j^{\mathrm{out}}}. \qquad (14)$$

It is straightforward to show that the stationary solution of the process is $\pi(\overrightarrow{jk}) = 1/L$ if the graph is Eulerian ($\sigma_j^{\mathrm{out}} = \sigma_j^{\mathrm{in}}$ for all j). One thus recovers the well-known result that the stationary probability of finding a walker on a node is proportional to its degree in undirected networks. If the underlying network is strongly connected, this is the only stationary solution of the process. By using the fact that

$$P(j;t) = \sum_i P(\overrightarrow{ij};t), \qquad (15)$$

and summing over j in Supplementary Equation (14), we find that

$$P(k;t+1) = \sum_j P(j;t) \frac{A_{jk}}{\sigma_j^{\mathrm{out}}}, \qquad (16)$$

thus recovering the standard master equation for a random walk process, driven by the transition matrix $A_{jk}/\sigma_j^{\mathrm{out}}$.

If the standard random walk is ergodic on a graph, the memory model is also ergodic on the same graph for any value of r_2, r_3, and $r_{3<}$, as long as each parameter is strictly positive, such that no transition is forbidden by the bias. In systems where ergodicity is not verified, we use link teleportation as described above. The robustness of link teleportation under variations of the teleportation probability $1 - \alpha$[29] is clear, after noting that the stationary solution of the first-order Markov process is $\pi(\overrightarrow{jk}) = 1/L$ when each node of the physical network has the same in-degree and out-degree (e.g., if the network is undirected), independently of α.

We now turn to evaluating the effects of r_2, r_3, and $r_{3<}$ on PageRank. To do so, we use a perturbation analysis of α close to 0, where a local approximation of PageRank is valid. For the sake of simplicity, we consider the case of unweighted networks. We further assume that the graph is Eulerian, so that the known solution $\pi(\overrightarrow{jk}) = 1/L$ for $r_2 + r_3 + r_{3<} = 1/3$ can be used as a baseline. In this case, it is straightforward to show that the dominant contribution to the stationary solution is

$$\begin{aligned} \pi(\overrightarrow{jk}) &= \frac{1-\alpha}{L} + \frac{\alpha}{L} \sum_i A_{ij} \hat{p}(\overrightarrow{ij} \to \overrightarrow{jk}) + o(\alpha^2) \\ &= \frac{1-\alpha}{L} + \frac{\alpha}{L} \sum_i A_{ij} \frac{A_{jk} w(\overrightarrow{ij} \to \overrightarrow{jk})}{\sum_l A_{jl} w(\overrightarrow{ij} \to \overrightarrow{jl})} + o(\alpha^2). \end{aligned} \qquad (17)$$

Intuitively, the PageRank of memory node $\overrightarrow{jk}$ increases when the bias of the random walk process favours transitions $\overrightarrow{ij} \to \overrightarrow{jk}$ over other transitions leaving $\overrightarrow{ij}$. Different types of scenarios are possible. As an illustration, we first focus on the role of r_2 for a network without triangles, such as the one of Supplementary Fig. 8a. As expected, higher values of r_2 tend to favour reciprocated links. Higher-order contributions, corresponding to paths of length longer than 1, are expected to favour reciprocated links connected to many other reciprocated links, etc., due to the iterative nature of PageRank. For instance, for memory node $\overrightarrow{12}$, one finds

$$\pi(\overrightarrow{12}) \;=\; \frac{1-\alpha}{8} + \frac{\alpha}{8}\left(\frac{1}{2} + \frac{8/10}{8/10+1/10}\right) + o(\alpha^2)$$

$$= \frac{1}{8}(1+\frac{7\alpha}{18})+o(\alpha^2), \quad (18)$$

as shown in Supplementary Fig. 9. For $\overrightarrow{21}$, in contrast, the linear approximation is uniform

$$\pi(\overrightarrow{21}) = \frac{1}{8}+o(\alpha^2), \quad (19)$$

and higher-order contributions are necessary to show that the bias boosts the PageRank. From (11) in the main manuscript, we see that physical nodes are important if they have many central incoming links. In the first example of Supplementary Fig. 8, node 2 is very central while node 4 is not. In Supplementary Fig. 8b, we also illustrate the role of r_2 on centrality, and show that high values of r_2 tend to favour links belonging to many triangles, as expected.

Supplementary References

1. Evans, T. & Lambiotte, R. Line graphs, link partitions, and overlapping communities. *Phys. Rev. E* **80**, 016105 (2009).
2. Ahn, Y., Bagrow, J. & Lehmann, S. Link communities reveal multiscale complexity in networks. *Nature* **466**, 761–764 (2010).
3. Bohlin, L., Esquivel, A. V., Lancichinetti, A. & Rosvall, M. Robustness of journal rankings by network flows with different amounts of memory. *Preprint at arXiv:1405.7832* (2014).
4. Liljeros, F., Giesecke, J. & Holme, P. The contact network of inpatients in a regional healthcare system. A longitudinal case study. *Math. Popul. Stud.* **14**, 269–284 (2007).
5. Taxi data. http://www.infochimps.com/datasets/uber-anonymized-gps-logs (2013).
6. Enron data. http://www.cs.cmu.edu/~enron (2013).
7. Efron, B. & Tibshirani, R. *An introduction to the bootstrap*, vol. 57 (Chapman & Hall/CRC, 1994).
8. Van der Heyden, M., Diks, C., Hoekstra, B. & DeGoede, J. Testing the order of discrete Markov chains using surrogate data. *Physica D* **117**, 299–313 (1998).
9. Clopper, C. & Pearson, E. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika* **26**, 404–413 (1934).
10. Fortunato, S. Community detection in graphs. *Physics Reports* **486**, 75–174 (2010).
11. Rosvall, M. & Bergstrom, C. Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci. USA* **105**, 1118 (2008).
12. Simonsen, I., Astrup Eriksen, K., Maslov, S. & Sneppen, K. Diffusion on complex networks: a way to probe their large-scale topological structures. *Physica A* **336**, 163–173 (2004).
13. Delvenne, J., Yaliraki, S. & Barahona, M. Stability of graph communities across time scales. *Proc. Natl. Acad. Sci. USA* **107**, 12755–12760 (2010).
14. Esquivel, A. & Rosvall, M. Compression of flow can reveal overlapping-module organization in networks. *Phys. Rev. X* **1**, 021025 (2011).
15. Rosvall, M., Axelsson, D. & Bergstrom, C. The map equation. *Eur. Phys. J. Spec. Top.* **178**, 13–23 (2009).
16. Lancichinetti, A. & Fortunato, S. Community detection algorithms: a comparative analysis. *Phys. Rev. E* **80**, 056117 (2009).
17. Rosvall, M. & Bergstrom, C. Mapping change in large networks. *PLoS ONE* **5**, e8694 (2010).
18. Edler, D. & Rosvall, M. The MapEquation software package. http://www.mapequation.org (2013).
19. Palla, G., Derényi, I., Farkas, I. & Vicsek, T. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* **435**, 814–818 (2005).
20. Kareiva, P. & Shigesada, N. Analyzing insect movement as a correlated random walk. *Oecologia* **56**, 234–238 (1983).
21. Bovet, P. & Benhamou, S. Spatial analysis of animals' movements using a correlated random walk model. *J. Theor. Biol.* **131**, 419–433 (1988).
22. Bergman, C. M., Schaefer, J. A. & Luttich, S. Caribou movement as a correlated random walk. *Oecologia* **123**, 364–374 (2000).
23. Pirolli, P. L. & Pitkow, J. E. Distributions of surfers' paths through the world wide web: Empirical characterizations. *World Wide Web* **2**, 29–45 (1999).
24. Chierichetti, F., Kumar, R., Raghavan, P. & Sarlós, T. Are web users really markovian? In *Proceedings of the 21st international conference on World Wide Web*, 609–618 (ACM, 2012).
25. Fronczak, A. & Fronczak, P. Biased random walks in complex networks: The role of local navigation rules. *Phys. Rev. E* **80**, 016107 (2009).
26. Burda, Z., Duda, J., Luck, J. & Waclaw, B. Localization of the maximal entropy random walk. *Phys. Rev. Lett.* **102**, 160602 (2009).
27. Boldi, P. & Rosa, M. Arc-community detection via triangular random walks. In *Web Congress (LA-WEB), 2012 Eighth Latin American*, 48–56 (IEEE, 2012).
28. Akaike, H. Information theory and an extension of the maximum likelihood principle. In *Selected Papers of Hirotugu Akaike*, 199–213 (Springer, 1998).
29. Lambiotte, R. & Rosvall, M. Ranking and clustering of nodes in networks with smart teleportation. *Phys. Rev. E* **85**, 056107 (2012).