Using the Hilbert curve

Cite as: AIP Conference Proceedings **707**, 388 (2004); https://doi.org/10.1063/1.1751382
Published Online: 08 June 2004

John Skilling

Using the Hilbert curve

John Skilling

Maximum Entropy Data Consultants Ltd, Killaha East, Kenmare, Kerry, Ireland

Abstract. The aim is to compute random samples from the posterior probability distribution for some object, modelled as a mixture distribution with a variable number of component "atoms", usually having relatively few attributes. We use a space-filling curve (specifically the Hilbert curve) to parameterise an atom's attributes by a single number, This simplifies the geometry, and we describe seven "engines" (LifeStory1&2, GuidedWalk, Leapfrog1&2, Chameleon1&2) for driving a MCMC exploration program. A binary variant of slice sampling underlies the engines.

INTRODUCTION

The defining feature of large problems is that they have a large number of parameters, viewed geometrically as a dimensionality. Large dimensionality has several awkward properties: most directions are nearly orthogonal to the one you want, the interesting domain you seek is usually exponentially tiny, and it is likely to have a peculiar shape as well. It is a lot harder to program in n dimensions than in 1. Yet n dimensions can be mapped to 1 by using a space-filling curve. Specifically, the Hilbert curve [1, 2, 3] uniformly covers the interior of a n-dimensional cube, whilst preserving a useful degree of locality. Using it, n coordinates can be encoded into a single number, albeit one with extended precision so that accuracy is preserved. Curves like this have been considered something of a curiosity, and have attracted only a few quirky applications [4, 5, 6, 7]. Indeed, to anyone classically trained in differential calculus and continuum mathematics, space-filling curves do look rather odd. However, from a computational point of view, using a Hilbert curve merely amounts to shuffling and re-defining the bits representing coordinate values, which is not particularly peculiar.

Of course, there is a price to pay. A function that is smooth and straightforward in two, three or more dimensions will look sharp and jagged when stretched out along a Hilbert curve. On the other hand, a function that is twisted and torn in several dimensions doesn't look appreciably worse. So, if we can work in one dimension at all (and we can), we may hope to be able to work with a wide class of functions that have traditionally been regarded as difficult. Indeed, I suggest that the ability to alter the dimensionality of a problem at will is a powerful and under-appreciated tool, which we should be able to use with advantage when exploring spaces of high dimension. Such an approach implies a change of focus, away from the lines and curvatures of geometry, and towards the connected-ness of topology.

CP707, *Bayesian Inference and Maximum Entropy Methods in Science and Engineering: 23rd International Workshop*, edited by G. Erickson and Y. Zhai

ATOMIC PRIORS

Another useful way of breaking down the complexity of a problem is to construct the object of interest as some number of *a-priori*-equivalent "atoms", which are scattered randomly over the domain of interest. These atoms are given whatever extra attributes are needed to model the object. Statisticians call the atoms in an atomic prior the "components of a mixture distribution" [8]. Atomic priors give a structure-based description. The number of atoms is usually allowed to vary, and the computer only needs to deal with the amount of structure that is actually required. There is no loss of generality: if required, we could let the whole object be a single atom with as many attributes as necessary. There are many applications where an atom has very few attributes, such as image reconstruction where an atom has only position (x,y), brightness, and possibly shape. It is then much easier to find acceptable and useful new attributes θ for an atom than it would be for the object as a whole, simply because of the huge reduction in dimensionality.

UNIT HYPERCUBE

Whatever the number (d) of attributes of an atom, it is always possible to squash the original prior into uniformity $\pi(\theta) = 1$ over the unit hypercube $[0,1]^d$. In one dimension, θ is the cumulant of the original prior. I recommend this discipline: there is no loss of generality and numerical exploration is likely to be easier. Also, there is no possibility of using an "improper" (non-normalised) prior, through assigning an infinite range or other pathology.

My convention, taking the computer word length to be W bits (usually 32) is to let the available coordinate values be odd multiples of $2^{-(W+1)}$, labelled by non-negative integers k from 0 to $2^W - 1$.

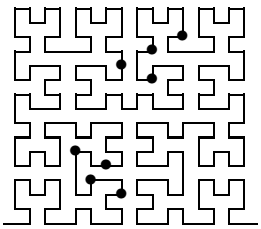
$$\theta = 2^{-W}(k + \tfrac{1}{2})$$

The rules of integer arithmetic (modulo 2^W) ensure that θ is wraparound continuous, which is never harmful and sometimes appropriate. This makes all states *a-priori*-equivalent geometrically as well as by prior measure, independently of any concern over the vagaries of floating-point representations. Helpfully, the states are symmetrically away from the boundaries $\theta = 0$ and $\theta = 1$, and centre $\theta = \frac{1}{2}$, which might be special in an application.

When θ has d dimensions (more than one), we can fill the unit hypercube with a space-filling Hilbert curve, thus reducing the topology to a one-dimensional coordinate along the curve. The hypercube contains $(2^W)^d$ digital points, so each point along the curve is labelled by an integer with Wd bits, or d words. A Hilbert curve preserves locality; a function that is continuous in space must also be continuous along the line (though not conversely). Winding a one-dimensional line into a d-dimensional volume necessarily places some points that are close in space far apart on the line. With the very precise and binary Hilbert pattern, there is at the extreme only one crossing of the central line $\theta_1 = \frac{1}{2}$ (vertical in the diagram), so that points on opposite sides of that line are mostly far apart on the Hilbert curve. Similarly, there is only one crossing

of the (wraparound continuous) abscissa $\theta_1 = 0$, and none at all of the base $\theta_2 = 0$. These barriers to free spatial movement are made harmless by moving them around occasionally, re-randomising the origin of the Hilbert curve within the hypercube and re-permuting its orientation.

One-dimensional representation along a line implies an ordering of locations, so that an atom's neighbours can be quickly identified by keeping a linked list. Neighbouring atoms along the line may not be quite the closest in space, but they still turn out to be useful because they identify pairs of atoms that may be similarly related to the data, and whose behaviour may thereby be correlated.



DETAILED BALANCE

We follow the usual Metropolis-Hastings approach [9, 10]with probabilistic transitions from state i to state j being set up to obey detailed balance

$$T_{ji}/T_{ij} = \pi(j)/\pi(i) \quad \text{where} \quad T_{ji} \equiv \Pr(j \leftarrow i)$$

in accordance with the prior $\pi(\cdot)$. Repeatedly applying the algorithm to an arbitrary initial probability assignment $\mathbf{p}$ will yield $\mathbf{p} \rightarrow \mathbf{T}\mathbf{p} \rightarrow \mathbf{T}^2\mathbf{p} \rightarrow \mathbf{T}^3\mathbf{p} \rightarrow \cdots$. Eventually, $\mathbf{p}$ will converge to the principal eigenvector of $\mathbf{T}$ with greatest eigenvalue (which, because the components of $\mathbf{p}$ always sum to the same unit total, must be 1). Provided the algorithm is aperiodic (so that it doesn't just bounce) and "irreducible" (so that every state is eventually accessible from any other), this principal eigenvector is unique [11, 12].

When likelihoods $L(\cdot)$ are present, probabilistic acceptance of transitions according to the rule

$$\text{"Accept transition } \; j \leftarrow i \; \text{ if and only if } \; L(j) \geq \texttt{Uniform}\big(0, L(i)\big)\text{"},$$

where `Uniform` means a random sample from the uniform distribution over the quoted range, ensures that the outcome converges to the posterior $L\pi$ instead of merely to the prior π. The trial change in log-likelihood should be neither much less than 1 (for which movement is inefficiently small) nor much greater than 1 (when most trials are rejected). It is difficult to anticipate the magnitude of change before calculating it, so we need some way of tuning the range of our transitions. Slice sampling [13]accomplishes this in one dimension, which is all we need. I use a binary variant.

BINARY SLICE SAMPLING

Let the initial state, represented by a B-bit integer k from domain $\mathscr{D}_0$ (usually the full Hilbert line), have likelihood $L(k)$. Set an acceptance level

$$a \in \texttt{Uniform}(0, L(k))$$

and randomise all B bits of k to reach

$$j_0 = k \oplus \texttt{Uniform}[0, 2^B)$$

in domain $\mathscr{D}_0$. According to Metropolis-Hastings, we are entitled to accept any new trial state j whose likelihood exceeds a, provided it was generated symmetrically with the reverse transition $j \to k$ being just as probable as the forwards $k \to j$. Randomisation was indeed symmetric, so if the likelihood for j_0 is greater than a, accept it.

If not, halve the domain size to $\mathscr{D}_1 \subset \mathscr{D}_0$ by randomising only the lowest $B-1$ bits,

$$j_1 = k \oplus \texttt{Uniform}[0, 2^{B-1}),$$

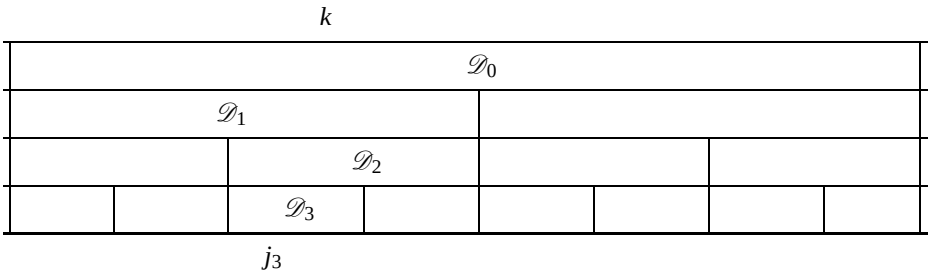
keeping the top bit intact. This transition too is symmetric because both k and j_1 lie in the same domain $\mathscr{D}_1$, so $j_1 \to k$ is just as likely as $k \to j_1$ (with probability $1/2^{B-1}$ as it happens). So, if the likelihood for j_1 is greater than a, accept it.

If not, halve the domain size again to $\mathscr{D}_2$ by randomising only the lowest $B-2$ bits,

$$j_2 = k \oplus \texttt{Uniform}[0, 2^{B-2}).$$

Again, this transition is symmetric, because k and j_2 both lie in $\mathscr{D}_2$ so $j_2 \to k$ is just as likely as $k \to j_2$, and each direction had the same chance of being aborted at the earlier stages. (This would no longer be true if j was being obtained by addition or subtraction instead of bit-randomisation, because k and j_2 would have approached each other via different ranges of j_1 and would hence have had different chances of aborting then: that's why I use the binary scheme.) So, if the likelihood for j_2 is greater than a, accept it.

If not, keep going by randomising fewer and fewer low-order bits, until an acceptable likelihood is reached. At worst, this procedure terminates after B trials with the original state k, necessarily acceptable because a was constructed to be beneath $L(k)$. More likely, an acceptable state will be found somewhere around the scale of the controlling likelihood function, at which $\Delta \log L = \mathscr{O}(1)$. All we ask of the likelihood is that it be a reasonably continuous function of position, so that there **is** an acceptable scale.



There appear to be barriers to free movement at special places such as $\theta = \frac{1}{2}$ (where passage between adjacent integers $0111111\cdots$ and $1000000\cdots$ requires all $B-1$ low bits to change). These can be made harmless by carrying out the procedure with respect to a randomly offset origin.

Neal [13]introduced slice sampling with more general nested domains, but our scheme is appropriate with integer arithmetic. Actually, Neal started his scheme from some intermediate length scale, and could step outward by widening the domain as well as inward by shrinkage. In a multi-atom environment, that is less important. Each atom has left and right neighbours along the Hilbert curve, and their location will often give sensible limits on that atom's movement. A location beyond can be considered "out-of-range", and rejected at once, involving merely the trivial overhead of checking whether a trial location is in range.

The following fragment of pseudo-code shows the basic implementation [14] of binary slice-sampling. The entry state is represented by the Wd-bit integer k.

$b \leftarrow W \times d$	full number of bits B
$o \leftarrow \texttt{Uniform}[0, 2^b)$	random origin
$a \leftarrow \texttt{Uniform}\big(0, L(k)\big)$	level of acceptance probability
do{	loop
$\quad j \leftarrow \big((k-o) \oplus \texttt{Uniform}[0, 2^b)\big) + o$	trial position around k
$\quad b \leftarrow b-1$	shrink interval around k
}**while**(j is out-of-range **or** $L(j) < a$)	until j is in range and if so is acceptably probable

The fact that the position almost certainly changes (exit state $j \neq k$) more than justifies the mild extra cost of slice sampling, as opposed to straightforward Metropolis-Hastings rejection.

NUMBER OF ATOMS

A typical prior for the number n of atoms is Poisson

$$\pi(n) = e^{-\alpha}\alpha^n/n!\,, \quad \alpha = \text{const.}$$

and we seek an algorithm that faithfully targets any such prior. The natural unit of change is just one atom at a time, so the only non-zero transitions T_{ji} will be $j = i+1$ ("birth" of an atom), $j = i-1$ ("death" of an atom), and $j = i$ (no change). Multiple births or deaths, as implied by the general treatment of jump-diffusion [15] or reversible-jump dynamics [16], are likely to be less often acceptable, so I restrict attention to single birth or death. Detailed balance fixes the ratios between birth and death rates but leaves their overall magnitudes free.

I choose to let each atom decay with unit mean lifetime, so that the death rate is set as

$$T_{n-1,n} = n\,dt$$

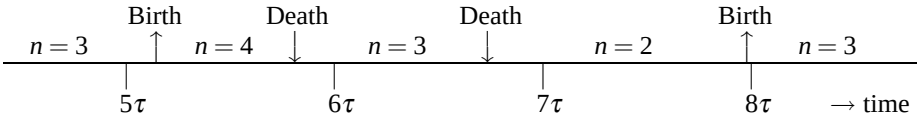
in infinitesimal interval dt of artificial time. The rationale for this is that most of the atoms will have been changed after unit time. Regular $\mathcal{O}(1)$ timesteps thus give a natural sampling period τ for our multi-atom object. Sampling more frequently would make successive objects tediously similar, whereas sampling less often might waste useful intermediate samples. Detailed balance for the Poisson prior implies a corresponding birth rate (β)

$$T_{n+1,n} \equiv \beta_n \, dt \,, \qquad \beta_n = (n+1)\, \pi_{n+1}/\pi_n = \alpha \,.$$

Starting with (say) n atoms, the time to the next event is exponentially distributed

$$\Pr(\Delta t) = (\beta_n + n)\, e^{-(\beta_n + n)\, \Delta t}$$

and when that event occurs, it is either birth or death in ratio $\beta_n : n$. Thus, in the following example, samples are taken at uniform times $\ldots, 5\tau, 6\tau, 7\tau, 8\tau, \ldots$ when n happens to be $\ldots, 3, 3, 2, 2, \ldots$.



There is a more abstract review of dimension-changing methods like this in [17].

EXPLORATION ENGINES

All the engines operate in the environment of an ensemble of $\mathcal{N}$ objects, each being a sample from the posterior distribution. The LifeStory engines operate on just one object at a time, but the others can use two or even three. To control this, we consider the ensemble to be a single supersystem $\Theta = \{\theta_1, \theta_2, \ldots, \theta_{\mathcal{N}}\}$ with its ensemble prior

$$\pi(\Theta) = \pi(\theta_1)\pi(\theta_2)\cdots\pi(\theta_{\mathcal{N}})$$

and its ensemble likelihood

$$\mathcal{L}(\Theta) = L(\theta_1)L(\theta_2)\cdots L(\theta_{\mathcal{N}})$$

giving its ensemble posterior

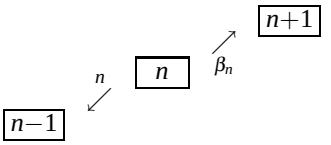
$$\mathcal{P}(\Theta) = P(\theta_1)P(\theta_2)\cdots P(\theta_{\mathcal{N}})$$

which factorises cleanly into the required posteriors of the constituent objects. Equilibrating just one object at a time is like exploring Θ by simple Gibbs sampling, but the supersystem allows more mixed exploration. Dimensionality need not be a curse; here it is an opportunity.

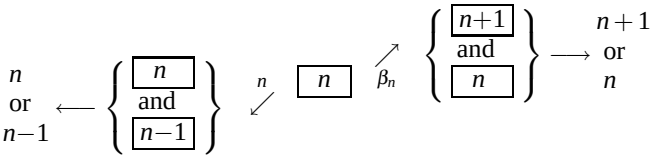
There is a reason for having several engines. Any individual engine can be thwarted by a particular form of likelihood. Even if provably convergent, it may be impractically slow. A different engine might overcome the defect, only to be thwarted elsewhere. The combination of all engines will only be defeated by likelihoods that defeat every one.

LIFESTORY1

The LifeStory1 engine operates on just one object in the ensemble, and combines the processes of birth, death, and movement – hence the name – in a rather natural way. First, we set up transitions for exploring n. When an event occurs, being birth or death in ratio $\beta_n : n$, the atomic number is to be incremented or decremented as appropriate, subject to Metropolis-Hastings acceptance to ensure detailed balance.



More slowly, but with equal validity, we can change the atomic number with probability 50%, leaving the indeterminate composite to be resolved afterwards.



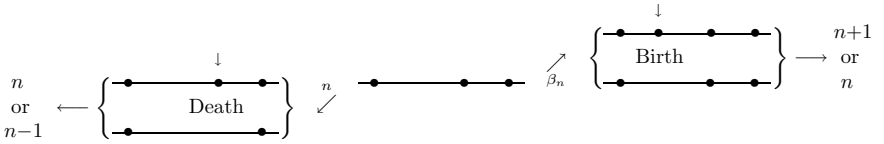
With likelihood factors L_j for j atoms, we proceed to sample from a birth composite, which has likelihood $L_{\text{birth}} = \frac{1}{2}(L_n + L_{n+1})$, according to the relative individual likelihoods:

$$\begin{aligned} \Pr(n+1 \mid \text{birth composite}) &= L_{n+1}/(L_n + L_{n+1}), \\ \Pr(\quad n \quad \mid \text{birth composite}) &= L_n \quad /(L_n + L_{n+1}). \end{aligned}$$

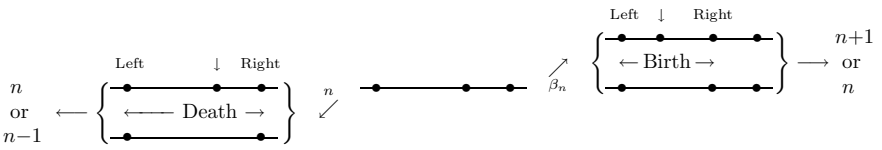
Likewise for a death composite of likelihood $L_{\text{death}} = \frac{1}{2}(L_n + L_{n-1})$:

$$\begin{aligned} \Pr(\quad n \quad \mid \text{death composite}) &= L_n \quad /(L_n + L_{n-1}), \\ \Pr(n-1 \mid \text{death composite}) &= L_{n-1}/(L_n + L_{n-1}). \end{aligned}$$

When atoms have attributes, birth of an atom is accompanied by selection of a random coordinate θ, and death is accompanied by selection of a random identifiable-by-location atom, as flagged by downward arrows in the diagram below.

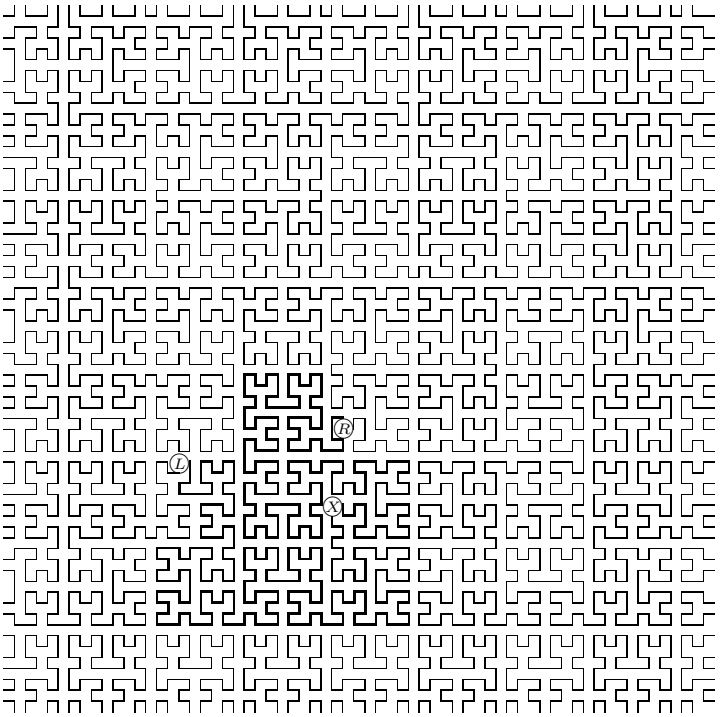


The likelihoods are promoted to functions, of position of the atom being born or killed in the presence of the other atoms, and the reason for introducing intermediate composite states now appears. The new position, whether that selected for birth or that of the atom under sentence of death, need not be immediately accepted or rejected by Metropolis-Hastings on the basis of that position only. Instead, movement is almost guaranteed by using slice sampling to adjust the composite state's position, using whichever of L_{birth} and L_{death} is appropriate. Slice sampling centres on the original selected position, and for efficiency it is carried out between the left and right neighbour atoms of that initial choice.



The birth or death of a complete atom might well cause significant change in the likelihood, and it is wise to couple the opportunity with a wider view of the possibilities by allowing movement. If a birth event succeeds in incrementing the number of atoms, all well and good. A change has been made, and slice sampling has likely improved the suggested location. If it fails, then there has been no change, which is regrettable. If a death event succeeds in decrementing the number of atoms, a change has been made. If not, then at least the selected atom will almost certainly have been moved. So, overall, LifeStory1 should make useful changes at least half the time.

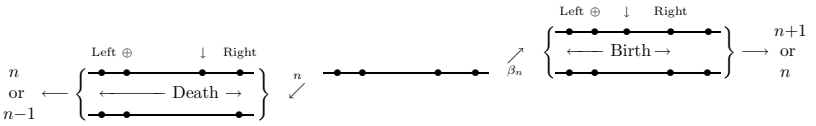
The illustration below illustrates the range allowed to a selected location X, constrained to move along the Hilbert curve (with origin randomised to $(5, 7)$ on a wraparound $2^6 \times 2^6$ grid) between its left and right neighbours L and R. With respect to the coordinates, this range is very roughly circular out to the distance of the neighbours.



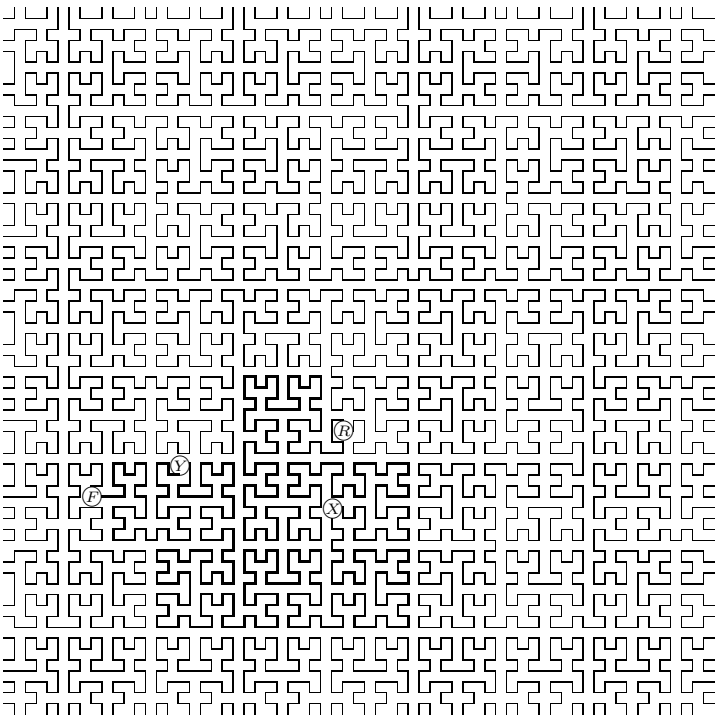
LIFESTORY2

The LifeStory2 engine, like LifeStory1, operates on just one object, and combines the processes of birth, death, and movement. The difference is that one of the neighbouring atoms is also allowed to move. Because they are indivisible changes, birth and death are often discriminated against by the associated changes in likelihood. The existing n atoms may have equilibrated as best they can, so that inserting or deleting a complete atom might always be unlikely if the other atoms are all fixed, even though global re-equilibration with $n \pm 1$ atoms might make the change acceptable or even preferable. In this situation, LifeStory1 is thwarted.

LifeStory2 allows a neighbouring atom to move aside to make room for the insertion, or move closer to compensate the deletion, so that the engine can jump around the barrier and equilibrate the number of atoms more effectively. In this way, a locally dominant constraint (such as mean position or total flux) can remain satisfied even while an atom is created or destroyed. "Splitting and combination moves" [18] are akin to LifeStory2, but restricted to using defined constraints.



The diagram shows the leftward neighbour "⊕" of the selected birth or death position "↓" being included in the process: inclusion of the rightward neighbour would be equally likely. In the upper state of either composite there are now two atoms that can be moved around, and in the lower state there is one. Slice sampling can equally well be carried out on two coordinates as on one. It centres on the original selected positions, and for efficiency it is done between the left and right neighbour atoms of those initial choices, as shown by the horizontal range arrows. Both *X* and its left neighbour (now called *Y*) can move between the further-left neighbour *F* and the original right neighbour *R*, allowing synergy between *X* and *Y* across a somewhat greater range.



If a birth event succeeds in incrementing the number of atoms, a good change has been made, with likely improvement to the suggested random location, and compensatory movement of a neighbour. Even if it fails, the neighbour will almost certainly have moved, so some change will have occurred. If a death event succeeds in decrementing the number of atoms, a neighbour will have moved to compensate, allowing the change to

be more sympathetic to the data (hence more favoured). If not, then at least the selected atom and a neighbour will almost certainly have both moved. So, overall, LifeStory2 should almost always make useful changes, and thereby should be at least twice as powerful as LifeStory1.

One can envisage generalising LifeStory2 by allowing yet more atoms to move. However, bringing in more atoms will expand the domain they cover, and may well bring in additional constraints. Slice sampling would have to work harder to find an acceptable pattern, and the movements would be correspondingly smaller. My judgment is that LifeStory2 is often likely to be about best.

GUIDEDWALK

The LifeStory engines explore a very roughly spherical domain about one or two selected positions. Especially in many dimensions, where the likelihood function is increasingly capable of constraining an atom anisotropically, this can be inefficient. An atom is allowed to move only a short distance along strongly-constrained directions, and this restriction carries over to weakly-constrained directions if exploration is isotropic, making their exploration slow. As it happens, the number of attributes (dimensions) ascribed to an atom is often small, so the mere fact of using an atomic prior much reduces the difficulty: we seek to control only one or two atoms at a time, not the entire object. Even so, it may be useful to avoid the inefficiency by using the local shape of the likelihood function. In "classical analogue" style, high-order methods such as conjugate gradient for maximisation, and hybrid Monte Carlo [19, 20] for probabilistic exploration, come to mind. These methods use a sequence of intermediate evaluations (of gradient as well as value of likelihood) to discover and incorporate the local shape.

Alternatively, we could simply use the ensemble. Let X be a randomly-selected atom that we wish to move, in accordance with the local likelihood function. We may presume that the existence of X at its particular location already represents some feature of the likelihood, in which case other objects should also have atoms in similar location representing the same feature. Let L and R be left and right neighbour atoms of the position of X, drawn from different objects. If, indeed, each object contains just one atom for the feature in question, then that will lie fairly closely left of X along the Hilbert curve half the time, and fairly closely right of X the other half. So L from its object should be the appropriate corresponding atom about half the time, and so should R from its object. Even if the precise correspondence and symmetry are relaxed, there should remain an appreciable $\mathcal{O}(1)$ probability $\mathcal{P}$ that L and R have a similar environment to X. The offset vector $(\mathbf{R}-\mathbf{L})$ will then be obedient to the extent and shape of the local likelihood function. Hence it can be suggested as an appropriate increment for the position of X, yielding a new trial location

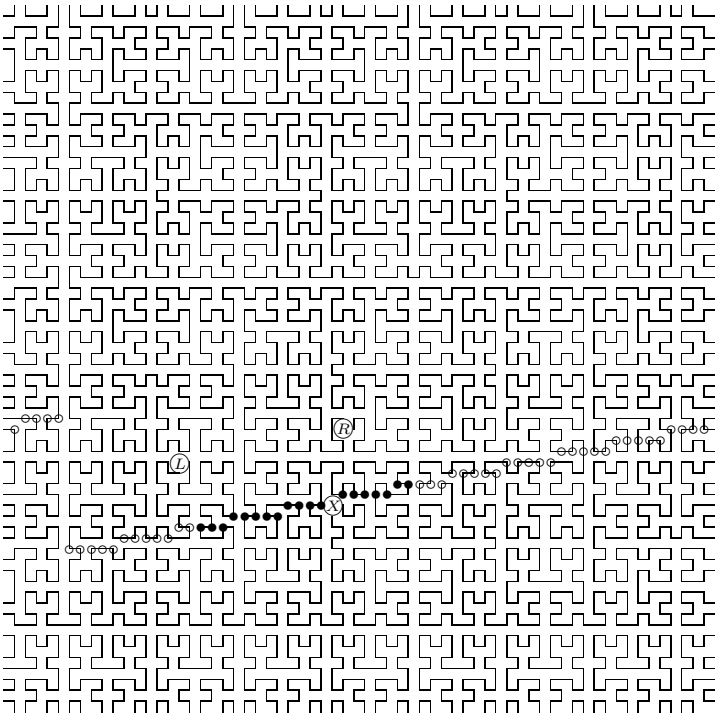
$$\mathbf{X}^{\text{trial}} = \mathbf{X} + s(\mathbf{R}-\mathbf{L}), \qquad s = \mathcal{O}(1).$$

Instead of a quasi-isotropic random walk, we have a **guided** walk in the direction $\pm(\mathbf{R}-\mathbf{L})$. As usual, the scale s of the change is difficult to predict, though we can guess $s \sim \nu^{-1/2}$ where ν is the number of locally active constraints. Larger changes

would usually be rejected by Metropolis-Hastings. Slice sampling keeps s nearly optimal though, whatever the restrictions.

Suppose there is one (or more) direction that is completely unconstrained, so that the likelihood function is extremely anisotropic. At any stage in the computation, the ensemble will have its corresponding atoms $X, L, R, \ldots$ distributed somehow along this axis, say with variance $\|\delta x\|^2 = \sigma^2$. After a step of the guided walk with $s = \nu^{-1/2}$, the variance of the newly accepted X^{trial} will be $(1+2s^2)\sigma^2 = (1+2/\nu)\sigma^2$, greater than before by a factor $1+2/\nu$. Hence the spread of atoms along the unconstrained direction(s) increases **exponentially**, with $\|\delta x\|$ gaining a factor something like $e^{\mathscr{P}}$ after ν ensemble-wide applications of the guided walk. Exponential growth occurs because the algorithm is invariant under affine transformation, so it automatically adjusts to whatever scales are operative (until supposedly-corresponding atoms become so distant that they can no longer be identified as such). It would be too optimistic to expect to recognise exponential behaviour in realistic applications. What we may see, though, is easy exploration of badly conditioned likelihood functions.

To implement GuidedWalk, we extend the vector $(\mathbf{R}-\mathbf{L})$ across the unit hypercube, starting at the arbitrary origin of the current space-filling Hilbert curve and continuing until the most-rapidly-changing coordinate (ξ say) increments by 1. On the digital B-bit grid, this defines a "staircase" of 2^B points parameterised by ξ. This is displaced along the less-rapidly-changing axes until it passes through $\mathbf{X}$, as shown by the circles in the illustration below (note the horizontal wraparound of 5 points caused by the Hilbert origin being randomised to $(5,7)$).



Filled circles identify those points of the staircase which lie on that part of the Hilbert curve for which L and R remain the left and right neighbours in their own objects. These are the trial points that are in detailed balance with X. To select one, we use slice sampling along the staircase, which almost guarantees acceptable movement along what is appreciably often a useful direction.

LEAPFROG1 AND LEAPFROG2

There are simpler ways of using the neighbours L and R of a selected atom X. Leapfrog1 uses just one leftward or rightward neighbour, L or R, preferably from an object different from X though it need not be. It takes

$$\text{either} \qquad \mathbf{X}^{(1)} = 2\mathbf{L} - \mathbf{X} \qquad \text{or} \qquad \mathbf{X}^{(1)} = 2\mathbf{R} - \mathbf{X}$$

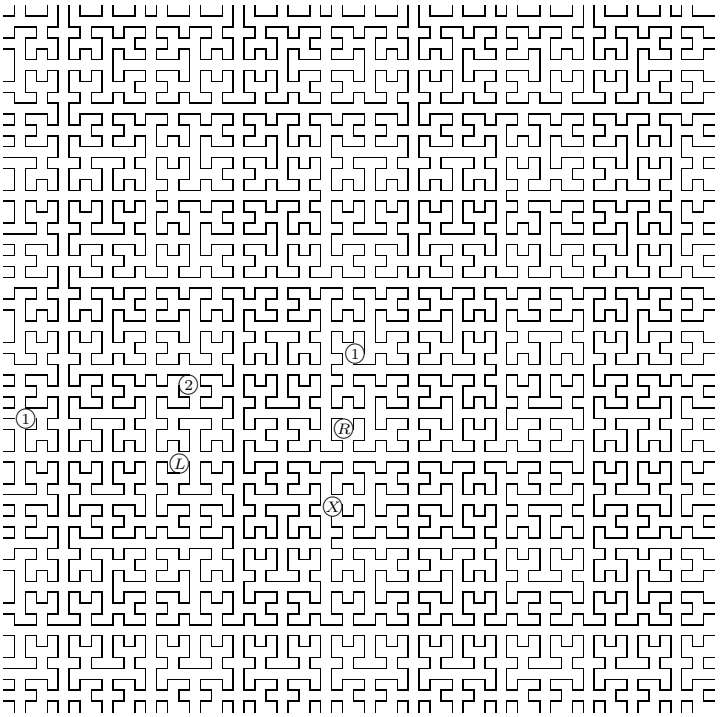
as a trial location, inverting X through L or R without any tunable coefficient. The trial location must be in detailed balance with X. Hence if L or R, in its member, is the left (right) neighbour of the location of X, then it must also be the right (left) neighbour of $X^{(1)}$ without any other intervening atoms. Whether this is likely depends on how atoms

are distributed locally. Provided this neighbourhood condition holds, the Metropolis-Hastings rule can be used to accept or reject the suggestion on the basis of its likelihood relative to X.

Leapfrog2 uses two neighbours, L to the left and R to the right, preferably drawn from objects other than X, but they could be both from the same as X. It takes

$$\mathbf{X}^{(2)} = \mathbf{L} + \mathbf{R} - \mathbf{X}$$

as a trial location, inverting X through the midpoint of L and R (which may be usefully closer to the centre of the local feature than either L or R individually). The trial location will be in detailed balance if L remains its leftward and R remains its rightward neighbour. Provided this neighbourhood condition holds, the Metropolis-Hastings rule can again be used to accept or reject the suggestion on the basis of its likelihood relative to X. In the illustration below, Leapfrog1 can invert X to either of the points marked ① , and Leapfrog2 can invert X to ② .



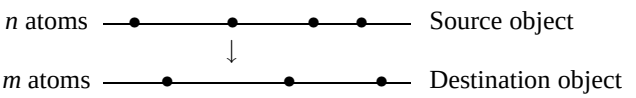
Like GuidedWalk, Leapfrog1 and Leapfrog2 are capable of increasing the spread of atoms along unconstrained directions **exponentially**. And, lacking the slice-sampling loop, they are simpler and faster to compute. However, if the atoms are subject to several constraints, the suggested changes will usually be rejected. Supposing that the local atoms are distributed in accordance with a locally Gaussian likelihood having ν constraints, each atom's chisquared misfit should be $\chi^2 \sim \nu$. But, according to the generat-

ing formulas, the misfits at the trial locations are likely to be larger: 5ν for Leapfrog1 and 3ν for Leapfrog2. The net effect is that the acceptance rate drops exponentially with the number of constraints, roughly like $e^{-0.4\nu}$ for Leapfrog1 and $e^{-0.25\nu}$ for Leapfrog2. This means that the Leapfrog engines can only operate well when the individual atoms have just a few constraints, though that will certainly be true if their number of attributes d is small.

No matter how long they are run for, the Leapfrog engines can only move atoms around the discrete lattice of points being integer combinations of the original positions. Whether or not this pattern of points is technically irreducible depends on Diophantine subtleties of whether the offsets in this lattice are co-prime in each dimension with the number of grid points, here 2^B. So the test of irreducibility may be less clear than one might suppose. In practice, we play safe and consider the Leapfrog engines to lack the irreducible property, resolving always to use them alongside irreducible engines like LifeStory which do provide full exploration.

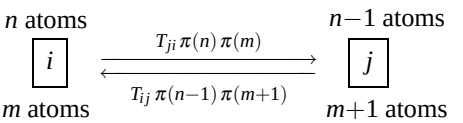
CHAMELEON1

Chameleon1 tries to make a randomly chosen atom jump from one object to another.



The idea is that an atom might be better placed in an object other than its source, especially if a useful patch of the hypercube is only just being discovered. Occasionally this could help the destination object out of a trap, so the engine could be worth including even if most of its transitions were rejected. Chameleon1 cannot be used on its own because its transitions are not irreducible: it can only use existing atom positions and cannot generate all the other possibilities. Hence it must be used with an irreducible engine such as LifeStory.

Care is needed with detailed balance. Let the original ensemble state "i" have n atoms in the source object and m in the destination. This occupancy has prior probability $\pi(n)\,\pi(m)$. The destination state "j" will have $n-1$ and $m+1$ atoms respectively, with prior probability $\pi(n-1)\,\pi(m+1)$. We require balance between the forward and backward transitions between these states.



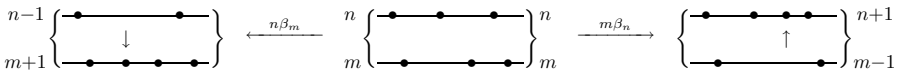
Hence the transition ratios must obey

$$\frac{T_{ji}}{T_{ij}} = \frac{\pi(n-1)\,\pi(m+1)}{\pi(n)\,\pi(m)} = \frac{n}{\beta_{n-1}}\,\frac{\beta_m}{m+1}$$

(remembering the birth and death rates for faithfully sampling the prior). This is implemented with a forwards rate

$$T_{ji} = n\beta_m\, dt$$

in infinitesimal interval dt of artificial time, implying the balancing backwards rate $T_{ij} = (m+1)\beta_{n-1}\, dt$. Starting with two objects with n and m atoms, jumps of an atom from n to m occur at rate $n\beta_m$, whereas jumps the other way from m to n occur at rate $m\beta_n$.



The time to the next event is exponentially distributed as

$$\Pr(\Delta t) = (n\beta_m + m\beta_n)\, \exp\big(-(n\beta_m + m\beta_n)\,\Delta t\big).$$

When that event occurs, it is either n-to-m or m-to-n in ratio $n\beta_m : m\beta_n$. Using an interval $\tau = \mathscr{O}(n+m)^{-1}$ of artificial time between observations offers each atom an $\mathscr{O}(1)$ chance of jumping. This is the natural "period" for which to run Chameleon1 between a given pair of objects, randomly selected from the ensemble.

A suggested jump is accepted or rejected on the basis of the ensemble likelihood

$$\mathscr{L} \;=\; \prod_{\text{objects}} L(\text{object})$$

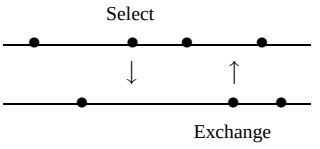
in which the only changeable factors are the likelihoods of the source and destination. According to the Metropolis-Hastings rule, a jump is accepted if

$$\mathscr{L}_{\text{new}} \geq \texttt{Uniform}(0, \mathscr{L}_{\text{old}})$$

and rejected otherwise.

CHAMELEON2

In the Chameleon2 engine, pairs of atoms exchange their ensemble membership. Equivalently, they exchange positions.



To encourage synergy between the jumps, the chosen atoms should be close together, so that the exchange atom should be a neighbour (in its own object) of the position of the atom first selected. Detailed balance is then trivially assured. The transitions do not affect the number of atoms in either object, so they stay faithful to the prior without

needing to adjust the relative rates. Suggested jumps are again accepted or rejected on the basis of the ensemble likelihood, according to whether or not

$$\mathscr{L}_{\text{new}} \geq \texttt{Uniform}(0, \mathscr{L}_{\text{old}}) \,.$$

As was done for Chameleon1, it seems appropriate for an iterate to operate the engine on a given pair of objects for long enough that each atom is offered an $\mathscr{O}(1)$ chance of jumping.

CONCLUSIONS

Of the seven engines described, LifeStory1&2 are always irreducible, so either can be used alone to enable full MCMC exploration. However, they navigate solely along the Hilbert line without reference to the original geometry, so will be inefficient in highly anisotropic applications. GuidedWalk does use the geometry, and its slice-sampling kernel uses the Hilbert line to identify neighbouring locations. However it is not irreducible: the number of atoms cannot change, and whether its geometrical exploration can reach all points depends on accidental features of the originating distribution. Hence GuidedWalk should only be used in conjunction with a LifeStory. Leapfrog1&2 and Chameleon 1&2 are simpler engines, again lacking irreducibility, that also need a LifeStory for completeness. The combination of engines gives an unconventional way of evolving an ensemble of objects by MCMC.

REFERENCES

1. Hilbert, D. (1891), "Über die stetige abbildung einer linie auf ein flächenstück", Mathematische Annalen **38**, 459–460.
2. Sagan, H. (1994) "Space-Filling Curves", Springer-Verlag, New York.
3. Skilling, J., (2004) "Programming the Hilbert Curve" (this volume).
4. Abend, K., Hartley, T.J. and Kanal, L.N. (1965) "Classification of binary random patterns", IEEE Trans. Information Theory **IT–11**, 538–544.
5. Bially, T. (1969), "Space-filling curves, their generation and their application to bandwidth reduction", IEEE Trans. Information Theory, **IT–15**, 658–664.
6. Stevens, R.J., Lehar, A. and Preston, F.H. (1983), "Manipulation and presentation of multidimensional images using the Peano scan", IEEE Trans. Pattern Analysis and Machine Intelligence,**5**, 520–526.
7. Song, Z. and Roussopoulos, N. (2002), "Using Hilbert curve in image storing and receiving", Information Systems, **27(8)**, 523–536.
8. Titterington, D.M., Smith, A.F.M. and Makov, U.E. (1985) "*Statistical analysis of finite mixture distributions* ", Wiley, Chichester.
9. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E. (1953) "Equation of state by fast computing machines", J. Chemical Physics **21**, 1087–1092.
10. Hastings, W.K. (1970) "Monte Carlo sampling methods using Markov chains and their applications", Biometrika **57**, 97–109.
11. Smith, A.F.M. and Roberts, G.O. (1993), "Bayesian computation via the Gibbs sampler and related Markov chain Monte Carlo methods", J. Roy. Statist. Soc. B**55**, 3–23.
12. Roberts, G.O. and Smith, A.F.M. (1994), "Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithm", Biometrika, **83**, 95–110.

13. Neal, R.M. (2003) “Slice sampling”, Annals of Statistics **31**, 705–767.
14. Skilling, J. and MacKay, D.J.M. (2003) “Discussion on slice sampling (by R.M. Neal)”, Annals of Statistics, **31**, 753–754.
15. Grenander, U. and Miller, M.I. (1994) “Representations of knowledge in complex systems (with discussion)”, J. Roy. Statist. Soc. B**56**, 549–603.
16. Green, P.J (1994), “Discussion on representations of knowledge in complex systems (by U. Grenander and M.I. Miller)”, J. Roy Statist. Soc. B**56**, 589-590.
17. Phillips, D.B. and Smith, A.F.M. (1996) “Bayesian model comparison via jump diffusions” in *Practical Markov chain Monte Carlo* eds. W.R. Gilks, S. Richardson and D.J. Spiegelhalter, Chapman and Hall, London.
18. Richardson, S, and Green, P.J. (1997) “On Bayesian analysis of mixtures with an unknown number of components” (with discussion). J. Roy. Statist. Soc. B**59**, 731–792.
19. Duane, S., Kennedy, A.D., Pendleton, B.J. and Roweth, D. (1987) “Hybrid Monte Carlo”, Phys. Lett. B**195**, 216–222.
20. Neal, R.M. (1993) “An improved acceptance procedure for the hybrid Monte Carlo algorithm”, J. Computational Physics **111**, 194–203.