Rendering: Importance Sampling

Bernhard Kerbl

Research Division of Computer Graphics
Institute of Visual Computing & Human-Centered Technology
TU Wien, Austria

With slides based on material by Jaakko Lehtinen, used with permission

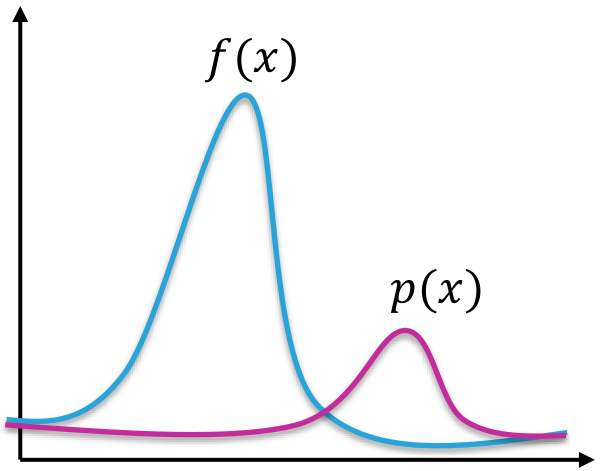
Today's Goal

- Improve the efficiency of Monte Carlo with importance sampling
- Understand how we can produce custom distributions in simple 1D, 2D and 3D domains by warping simple, uniform random variables
- Learn how we can transform samples between cartesian and non-cartesian domains (e.g., from polar (θ, ϕ) to XYZ vectors)
- Understand how we can incorporate these steps into path tracing

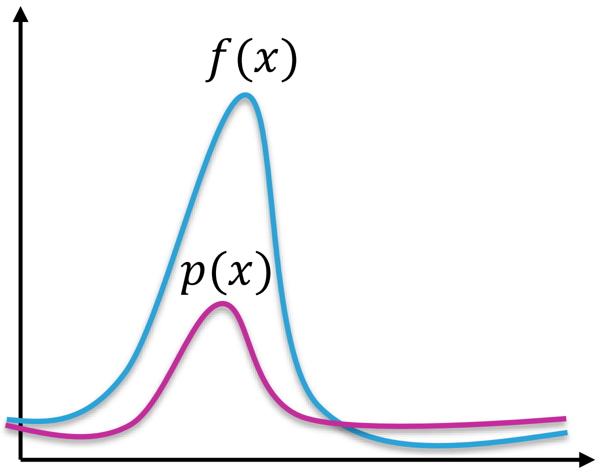
Importance Sampling

- All these things sound tedious… why do we need to create samples from arbitrary distributions? In different domains even?

- When we sample, e.g., the hemisphere, we can use any PDF we like



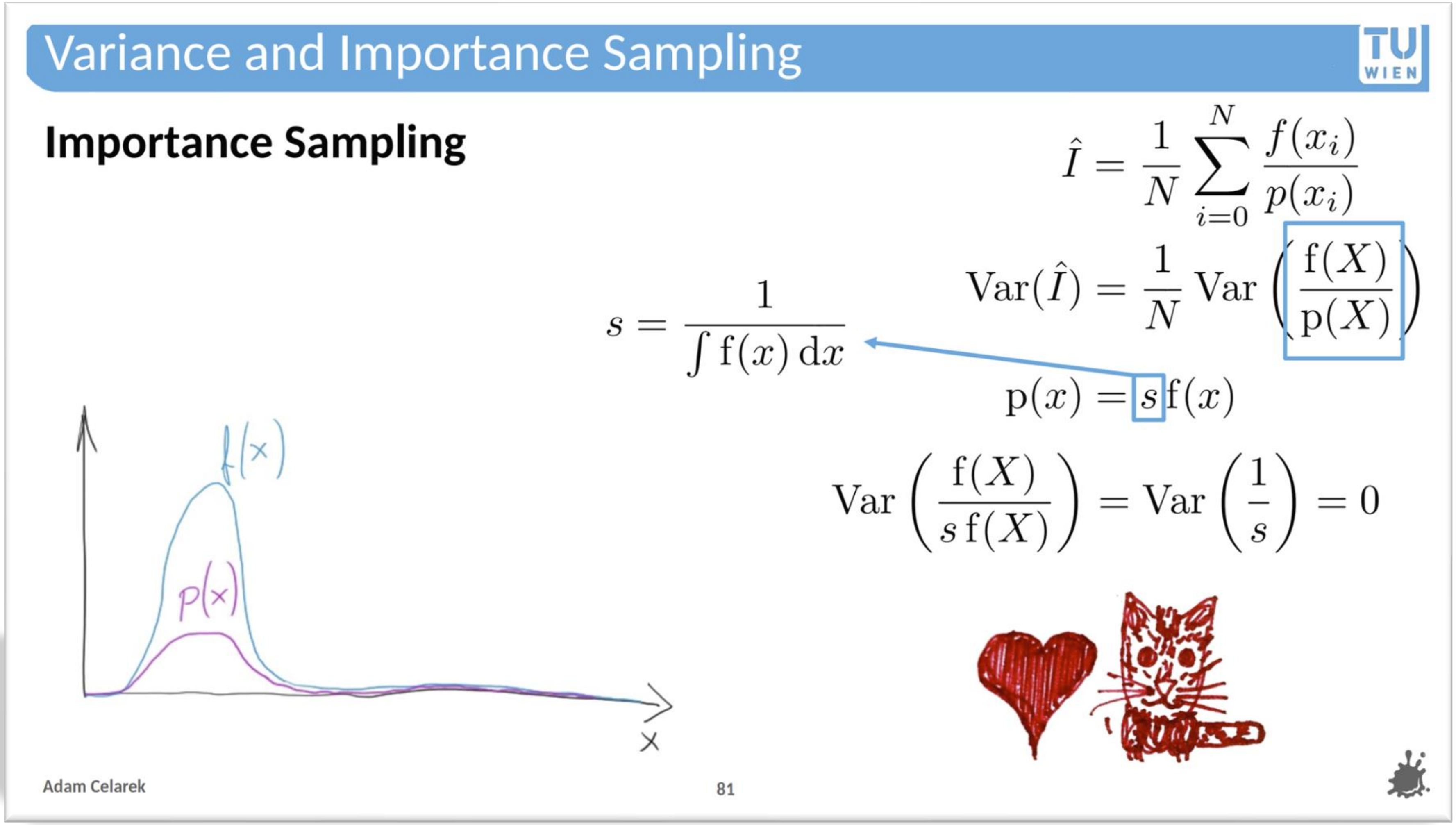
Bad sampling (high variance)



Importance sampling (low variance)

- We know the selection of the proper $p(x)$ as importance sampling

- Remember: if possible, you want a PDF $p(x)$ that mimics $f(x)$!



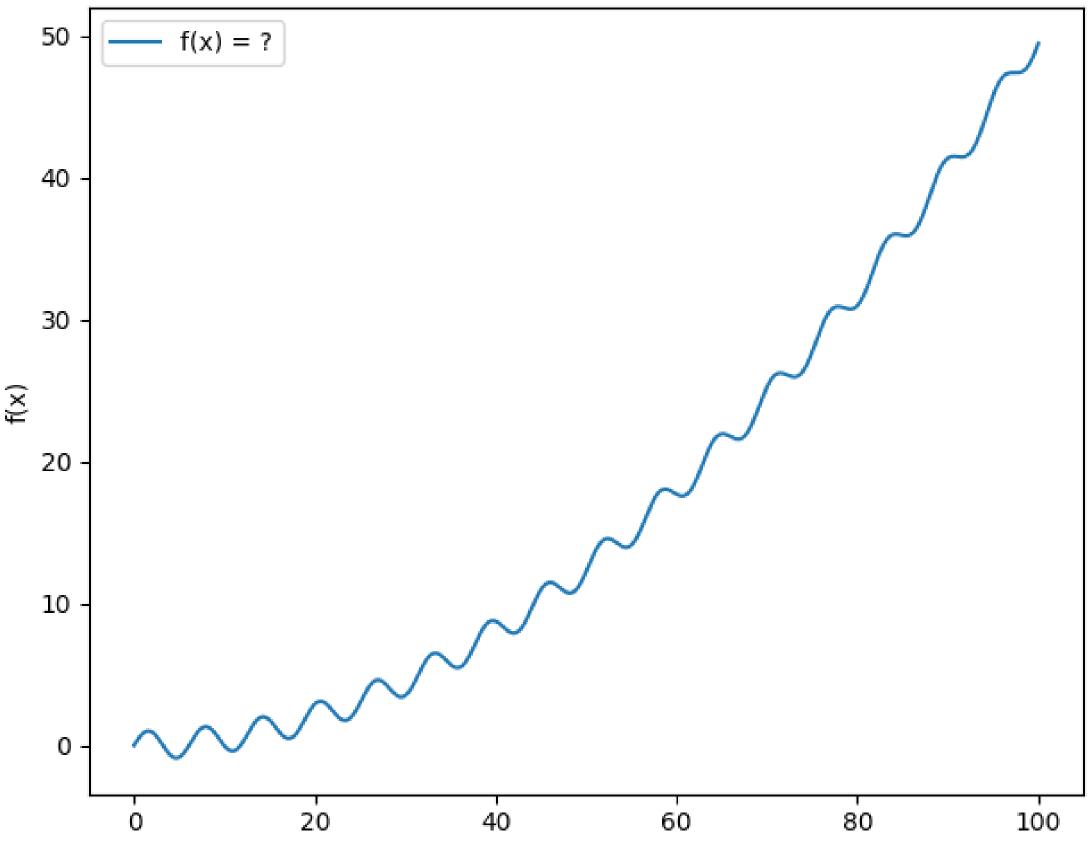
Monte Carlo Integration with Importance Sampling

- Let's look at an application for importance sampling in practice

- Consider a target function $f(x)$

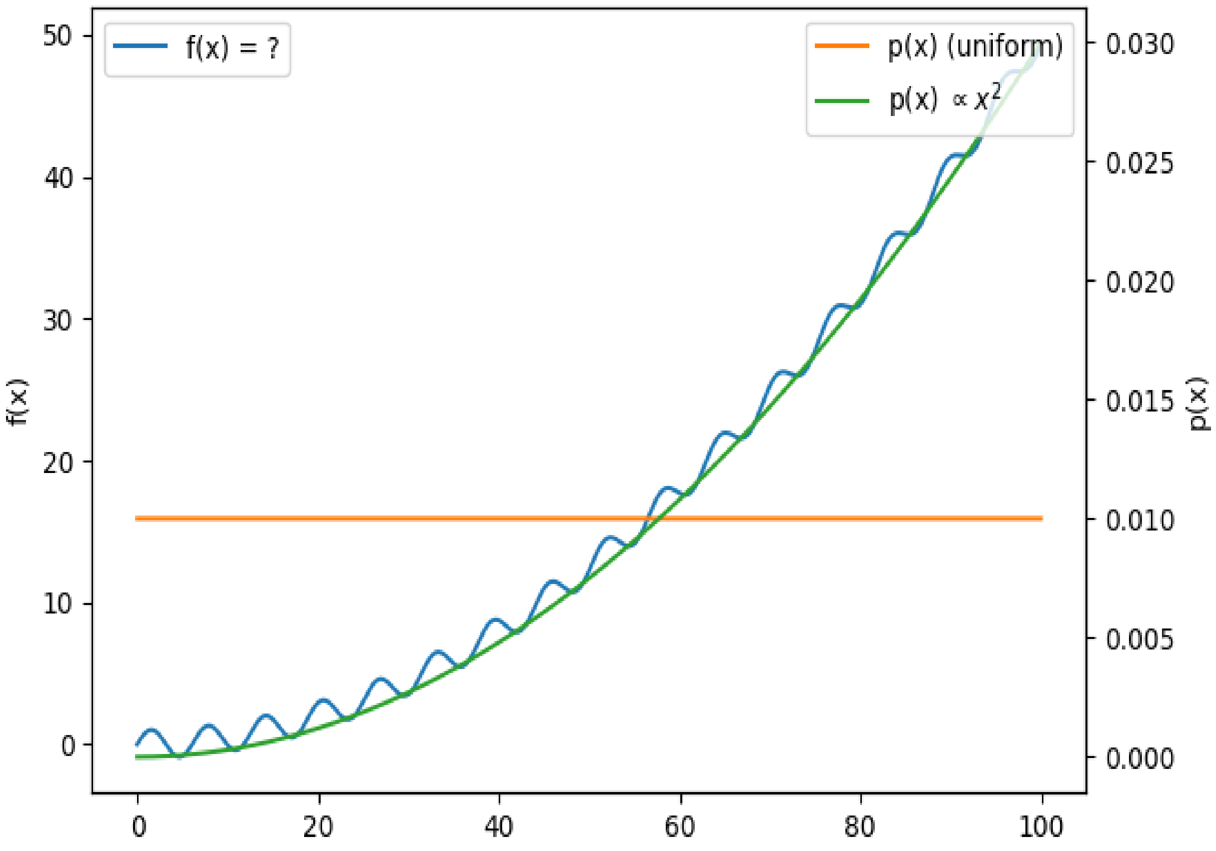
- You want to compute its integral, but have no closed-form solution or don't know what $f(x)$ is?

- Clearly, a case for Monte Carlo



Monte Carlo Integration with Importance Sampling

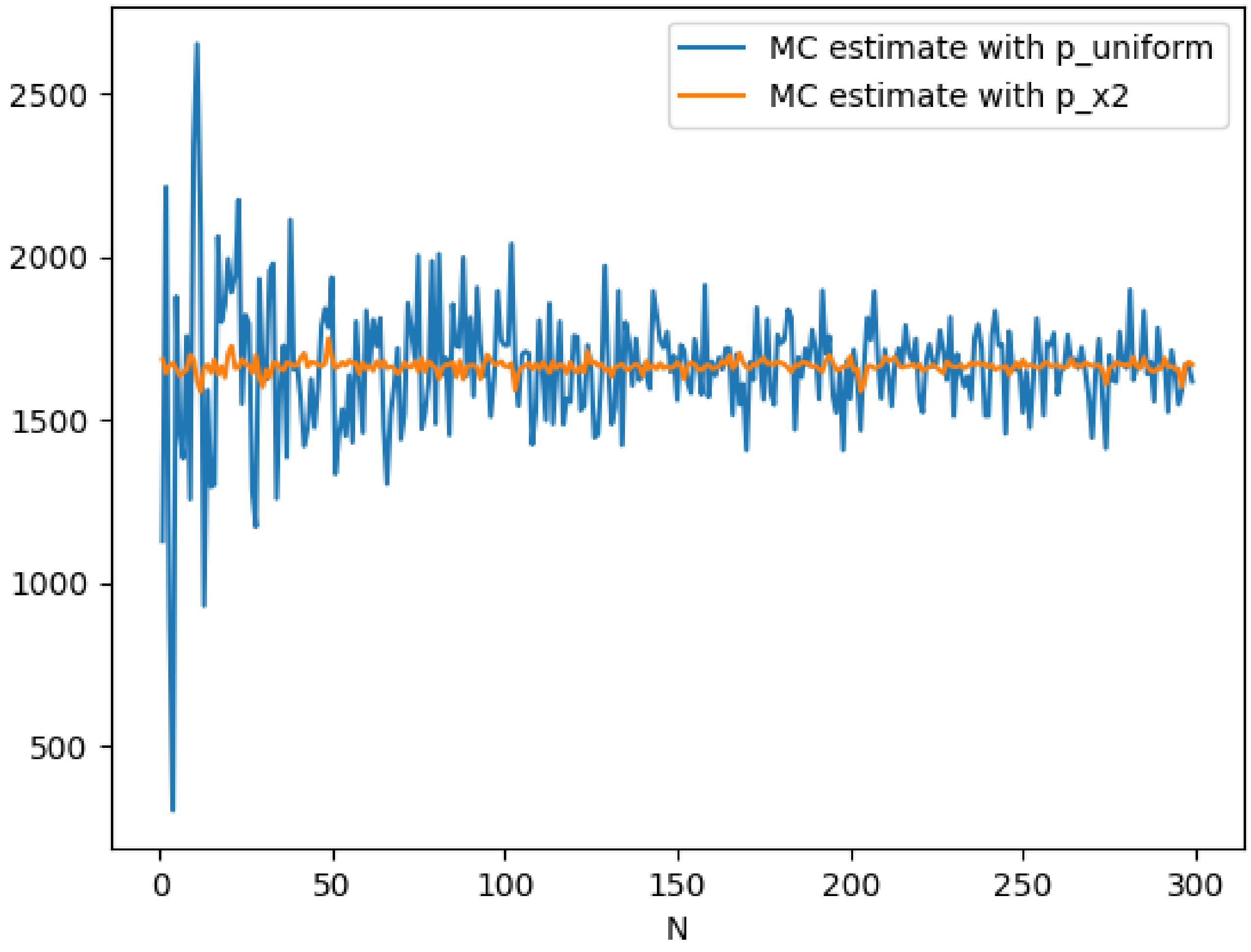
- If we take another look, the shape of this function seems familiar…
- It appears to be quite close to x^2!
- We already know that uniform sampling of $f(x)$ is only **one** way to do Monte Carlo integration…
- Let's try instead with $p(x) \propto x^2$



Uniform vs Importance Sampling (Python)

integrate_mc(0, 100, N, f, **p_uniform**, **gen_uniform**) *vs* integrate_mc(0, 100, N, f, **p_x2**, **gen_x2**)

- Both methods converge towards the same result
- But the importance-sampled method converges quicker!
- Let's see what the code behind it looks like..



Uniform vs Importance Sampling (Python)

```
integrate_mc(0, 100, N, f, p_uniform, gen_uniform)   vs   integrate_mc(0, 100, N, f, p_x2, gen_x2)
```

```python
def integrate_mc(a: float, b: float, N: int, f, p, gen):
    X = gen(a, b, N)
    estimates = f(X)/p(X, a, b)
    result = estimates.sum() / N
    return result

def p_uniform(x, a: float, b: float):
    return x/(b-a)

def p_x2(x, a: float, b: float):
    b3 = ((b**3)/3)
    a3 = ((a**3)/3)
    return x**2/(b3-a3)
```

?

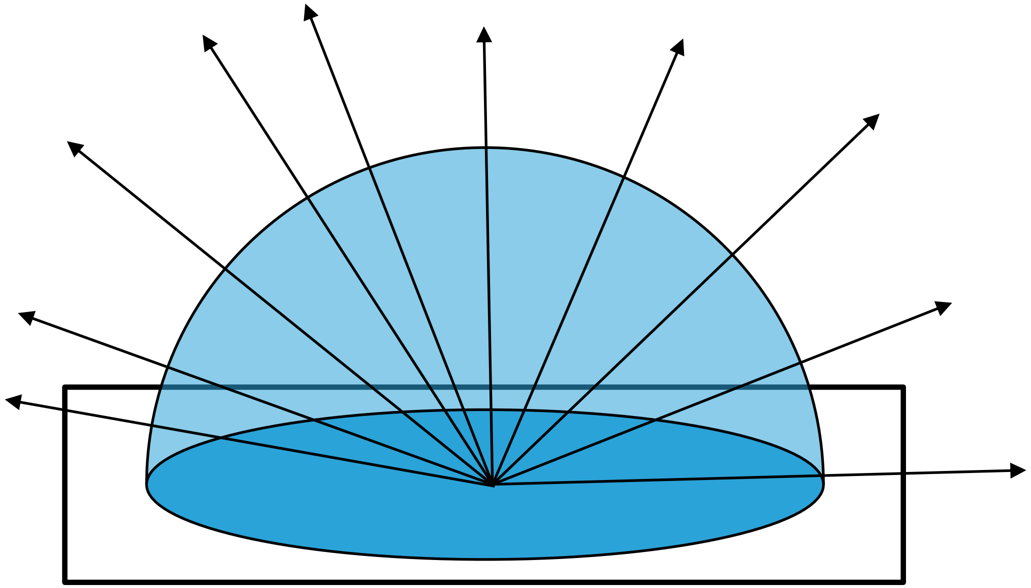
```python
def gen_uniform(a: float, b: float, N: int):
    xi = np.random.rand(N)
    return xi * (b - a) + a

def gen_x2(a: float, b: float, N: int):
    xi = np.random.rand(N)
    b3 = (b**3)
    a3 = (a**3)
    return (a3+xi*(b3-a3))**(1.0/3.0)
```

?

By the end of the day, this should make sense to you!

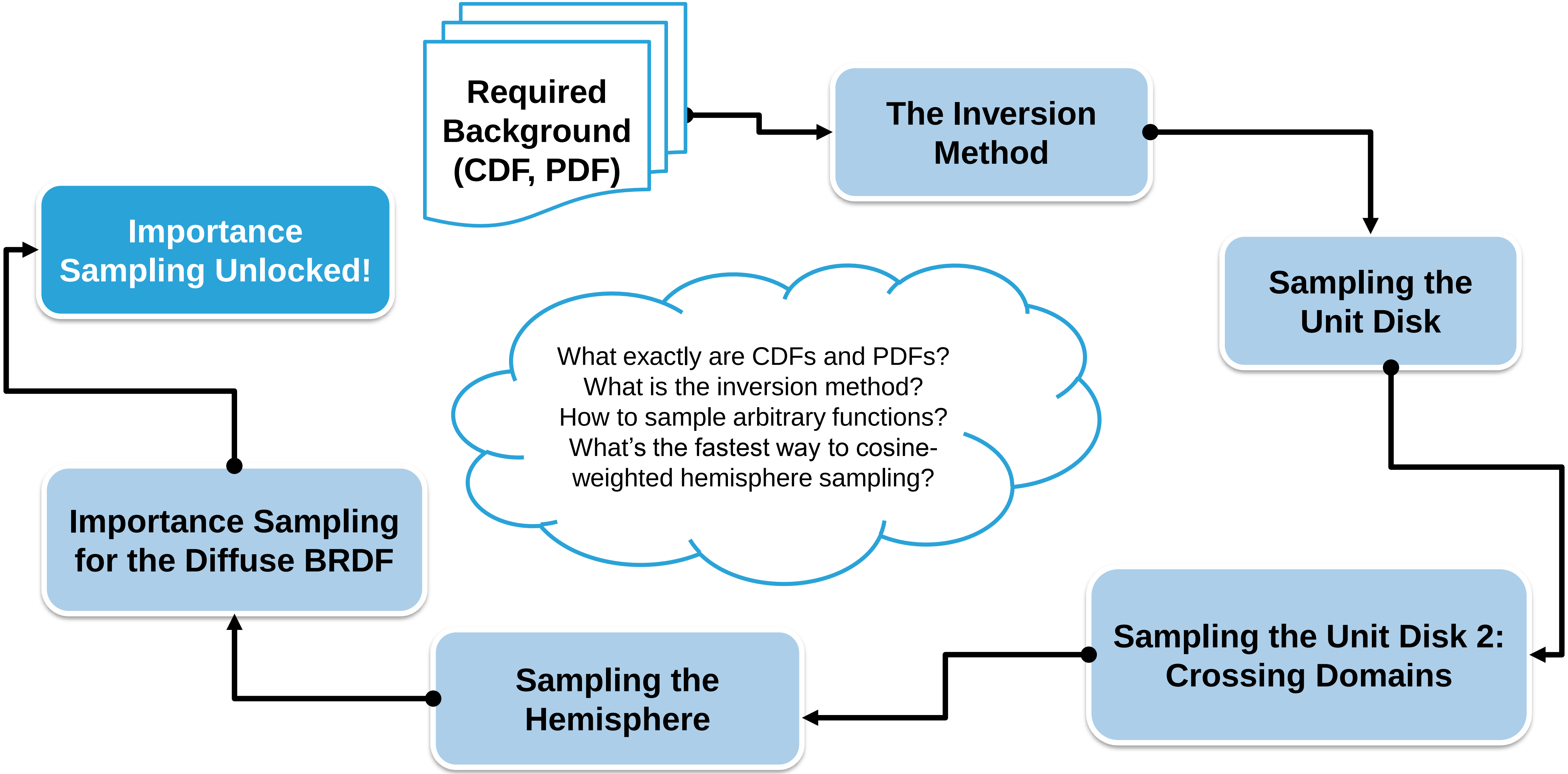
Importance Sampling on the Hemisphere

- Before, we did uniform hemisphere sampling, and it worked
- But perhaps we can also use importance sampling here?
- Can we perhaps importance-sample the *rendering equation*?

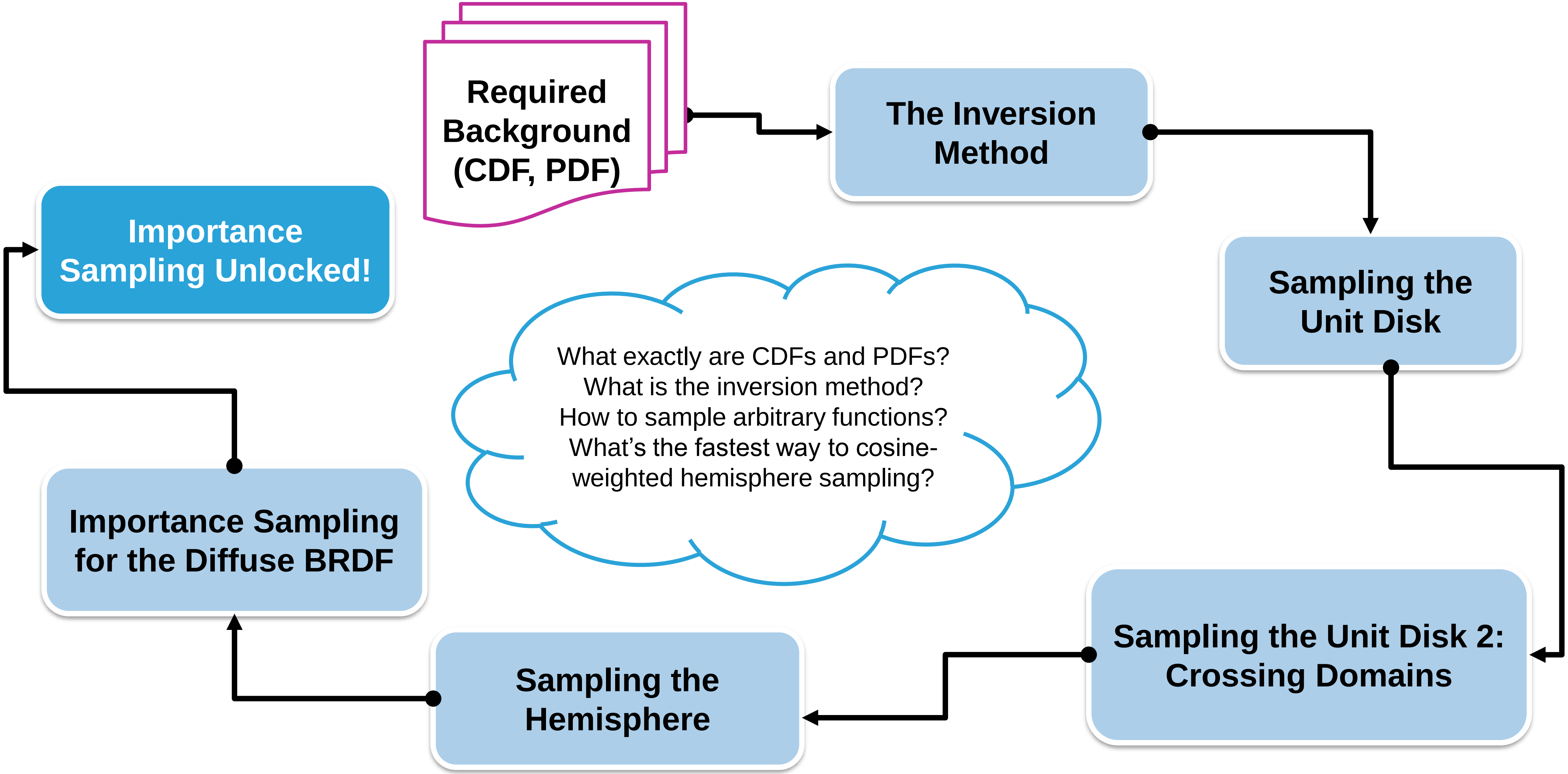


- The hemisphere is a peculiar domain. Sampling it with arbitrary distributions is a little bit more complex…

Today’s Roadmap



Today’s Roadmap



Discrete Random Variables

- In daily life, we are mostly confronted with *discrete* random results
  - A coin flip
  - Toss of a die
  - Cards in a deck

- Each possible outcome of a random variable is associated with a specific probability p. Probabilities must sum up to 1 (100%)

- E.g., a fair die: $X \in \{1,2,3,4,5,6\}$ and $p_1 = p_2 = \cdots = p_6 = \frac{1}{6}$

Continuous Random Variables

- A continuous random variable X with a given range $[a, b)$ can assume any value X_i that fulfills $a \leq X_i < b$

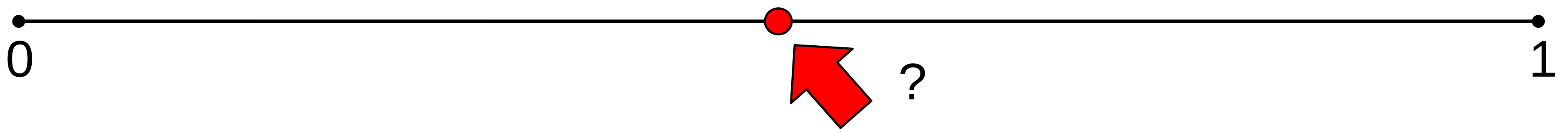
- Working with continuous variables generalizes the methodology for many complex evaluations that depend on probability theory

- There are infinitely many possible outcomes and, consequently, the observation of any specific event has with vanishing probability

- How can we find the probabilities for continuous variables?[2]

Cumulative Distribution Function (CDF)

- For continuous variables, we cannot assign probabilities to values



If X can take on any value with equal probability, what is the probability of $X = 0.5$?

- The *cumulative distribution function* (CDF) lets us compute the probability of a variable taking on a value *in a specified range* [2]

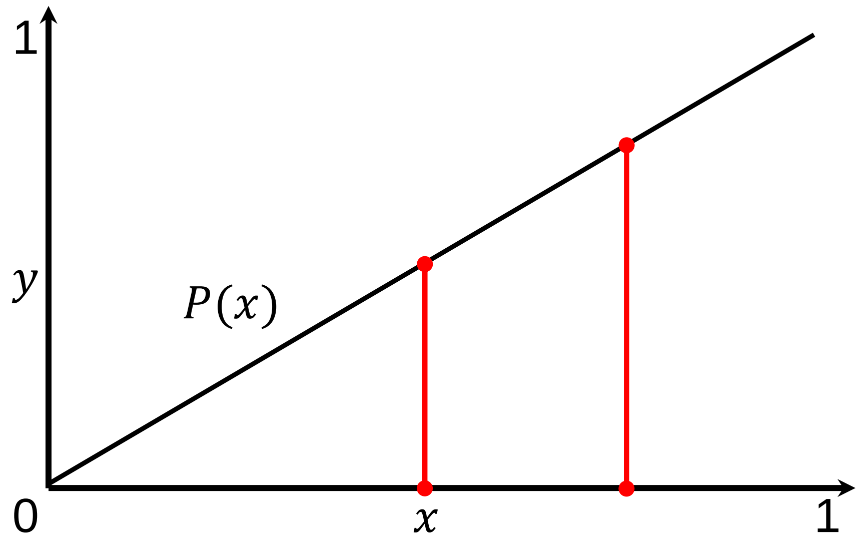
- We use notation $P_X(x)$ for the CDF of X's distribution, which yields the probability of X taking on any value $\leq x$

Probability for a Range with CDF

- $P_X(b) - P_X(a) \ = \ Pr\{a \leq X_i \leq b\}$

- Read as: $the\ probability\ of\ X\ taking\ on\ any\ value\ from\ 0\ to\ b,$ $minus\ the\ probability\ of\ X\ taking\ on\ any\ value\ from\ 0\ to\ a$

- Example: uniform variable ξ generates values in range [0, 1):
  - $P_\xi(x) = x$
  - $P_\xi(0.75) - \mathrm{P}_\xi(0.5) = 0.25$

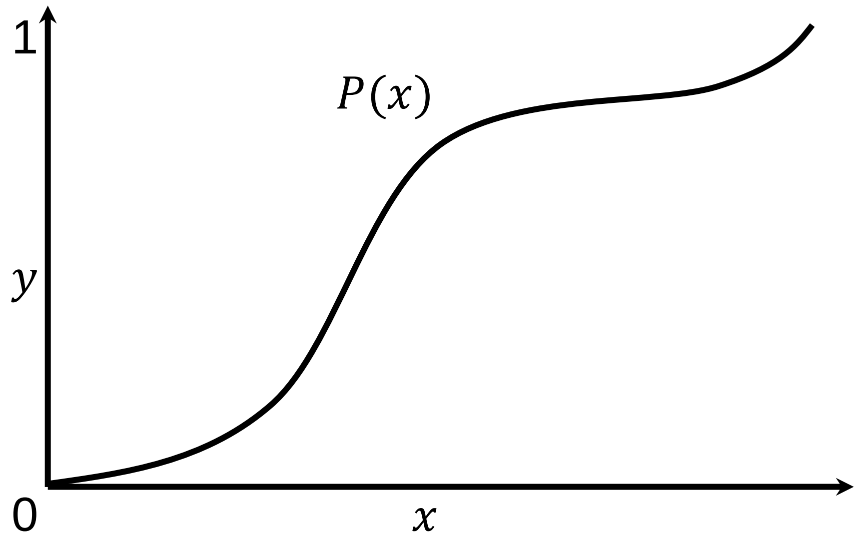


Properties of the CDF

- CDF is bounded by [0, 1] and monotonic increasing
  - Probability of **no** outcome is 0, the probability of **some** outcome is 1
  - Die: Rolling a number between 1 and 6 cannot be less probable than rolling a number between 1 and 5

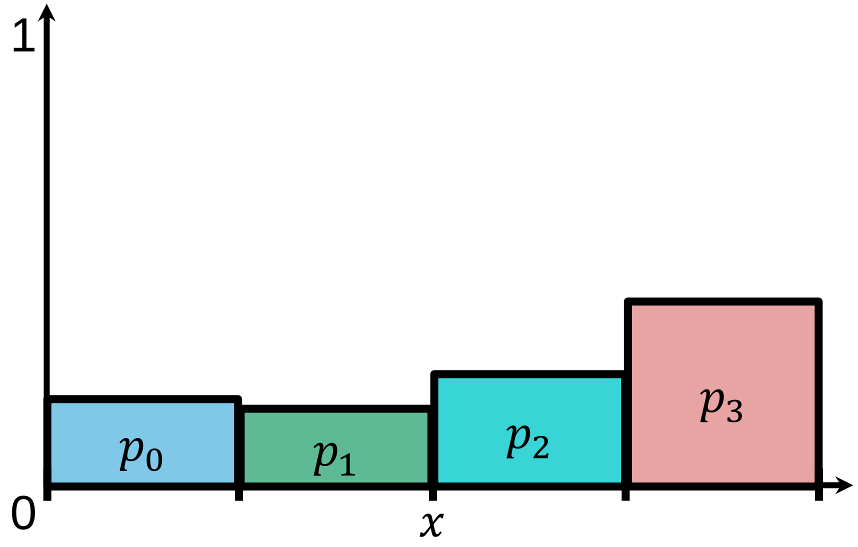
- CDFs can be applied for discrete and continuous random variables

- How do we compute the CDF?

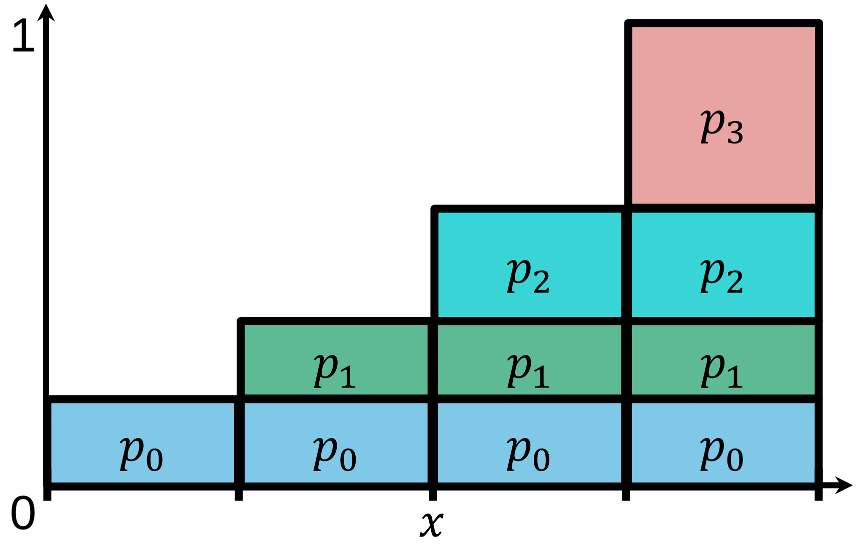


Computing the CDF for Discrete Random Variables

- Determine the limits $[a, b]$ of your variable X
- For each outcome, find its probability $p_a, \dots, p_b$
- The CDF of that variable is then a function $P_X(x) = \sum_{i=a}^{x} p_i$



Outcome Probabilities



Cumulated Probabilities (CDF)

Probability Density Function (PDF)

- The PDF $p(x)$ is the derivative of the CDF $P(x)$: $p(x) = \frac{dP(x)}{dx}$
- For a PDF $p(x)$, $P(x) = \int p(x)\, dx$ and $\int_a^b p(x)\, dx = P(b) - P(a)$
- $p(x)$ must be positive everywhere: a negative value would mean we can find $[a, b]$ such that $\int_a^b p(x)\, dx$ has a negative probability
- $p_X(x)$ can be understood as the **relative** probability of $X_i = x$.
  I.e., if $p_X(a) = 2p_X(b)$, then $X_i = a$ is twice as likely as $X_i = b$

Notes about the PDF

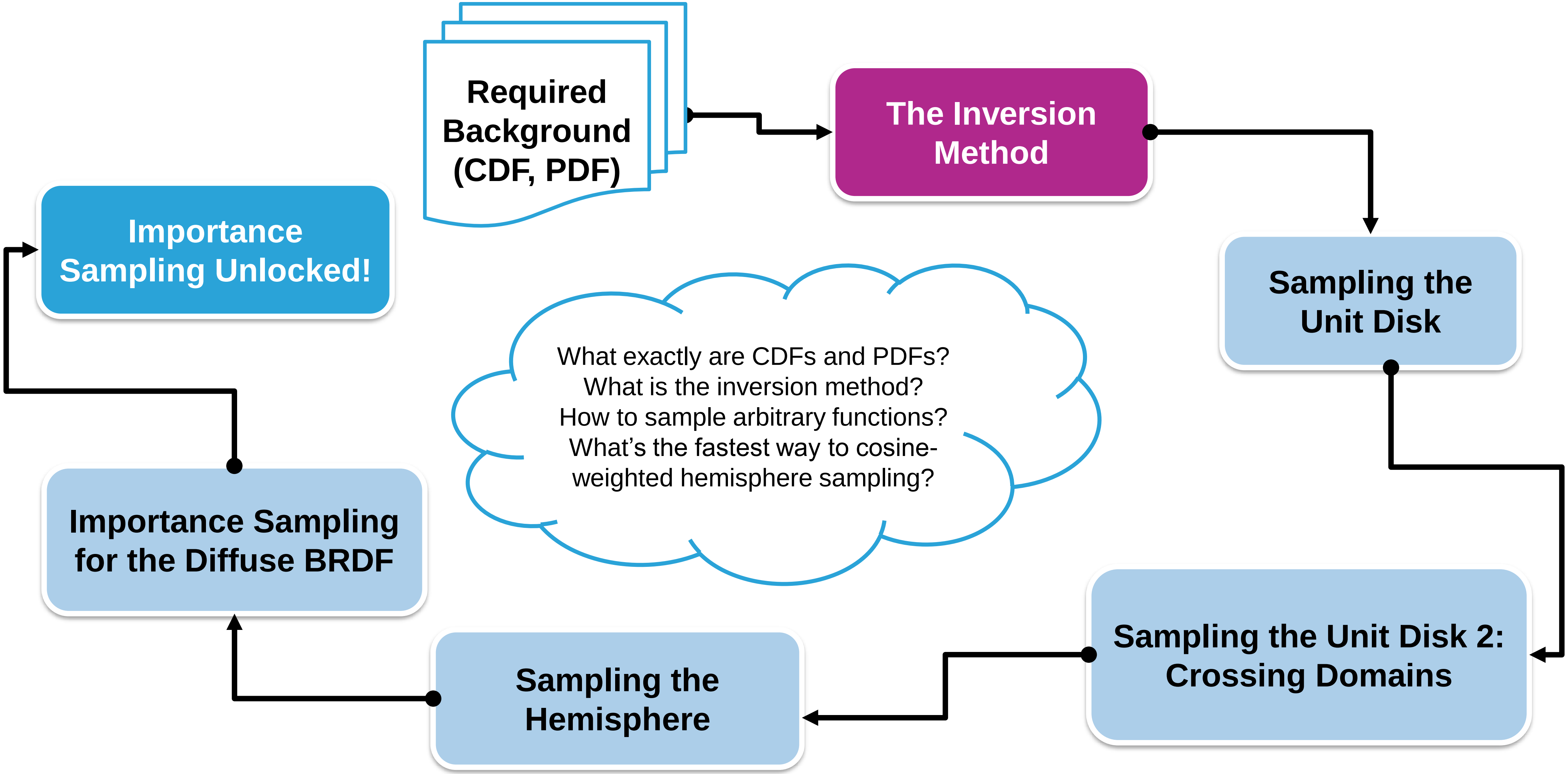
- Notation may **look** like probability, but PDF values can be >1!

- For both discrete and continuous variables, we can reference "$p(x)$" to describe their distribution

- **Summary**: for a continuous variable X with a known, integrable PDF, we can find the CDF and probabilities of X landing in a given *range*

- …is this actually helpful?

Today's Roadmap



Creating Variables with Custom Distributions

- By discovering the CDF, we have found a powerful new tool
- The function is often invertible: this means, we can generate random variables with a desired distribution!
- Rationale: Since the CDF is monotonic increasing, there is a unique value of $P_X(x)$ for every x with $p_X(x) > 0$
- More informally, if we plot a 1D CDF, any x value that X can take on has a unique, corresponding coordinate on the y-axis

Basic Sampling with Canonical Random Variables

- We want to generate samples for a custom distribution from a random input that we can easily obtain in a computer environment

- Our assumed input is the **canonical random variable** ξ:
  - continuous (i.e., a **real** data type)
  - with **uniform** distribution
  - in the range **[0, 1)**

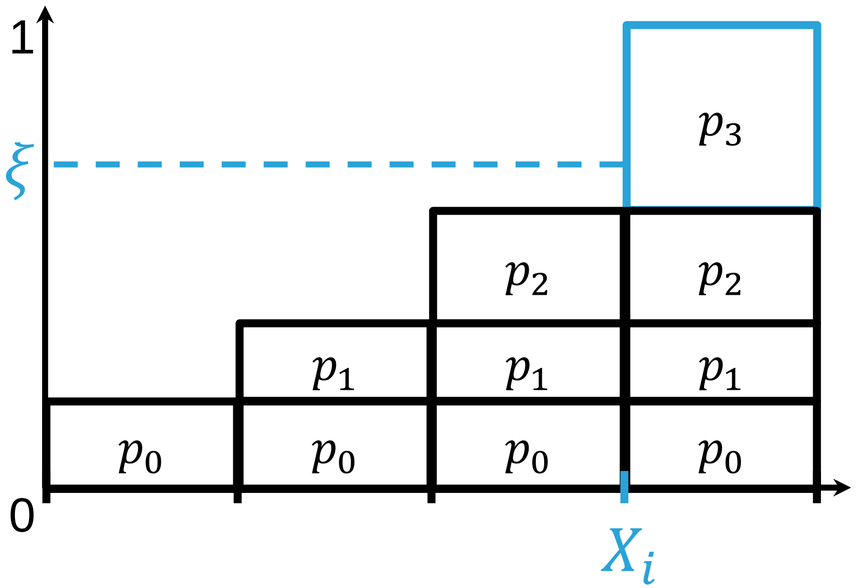
- Goal: warp samples of ξ to ones distributed according to some $p(x)$

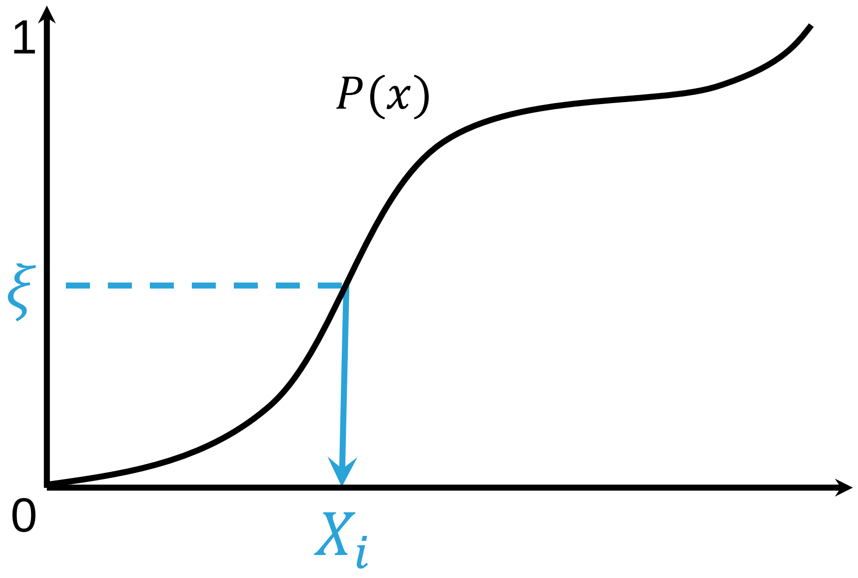
The Canonical Random Variable

- Our assumed default input variable
- PDF for ξ is 1 in range $[0,1)$ and 0 everywhere else
- CDF for ξ is linear
- Important property: we have equality $P(\xi_i) = \xi_i$

The Inversion Method

- For discrete variables: we draw ξ and iterate event probabilities
- When their sum first surpasses ξ, we have found X_i
- For continuous variables: exploit P_X's bijectivity and use its inverse!
- We can use canonic ξ to compute $X_i = P_X^{-1}(\xi)$ according to $p_X(x)$





Example: Exponential Distribution

- Used mainly for estimation of time intervals between two events

- The probability of a value decreases exponentially

- Needs additional parameter λ, often called *rate parameter*

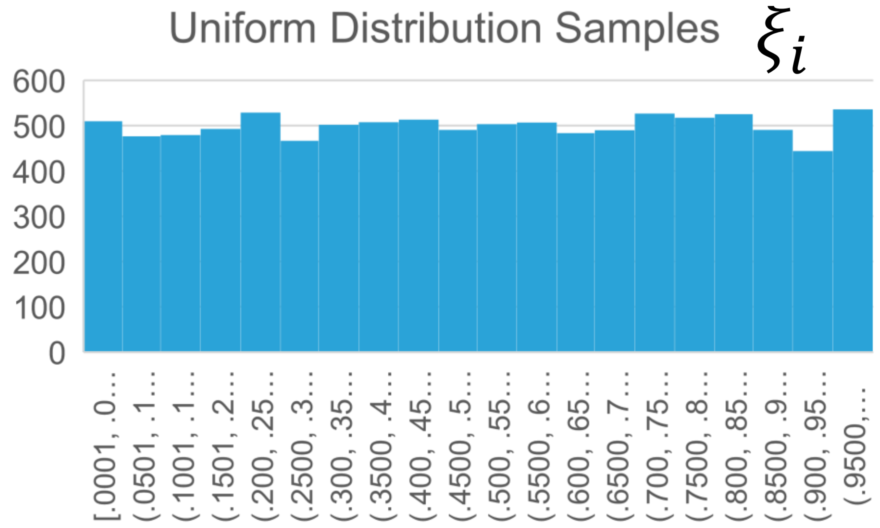
- We can find its PDF and CDF in most probability text books
  - $p(x, \lambda) = \lambda e^{-\lambda x}$
  - $P(x, \lambda) = 1 - \mathrm{e}^{-\lambda x}$, $P^{-1}(x', \lambda) = -\frac{\log(1-x)}{\lambda}$

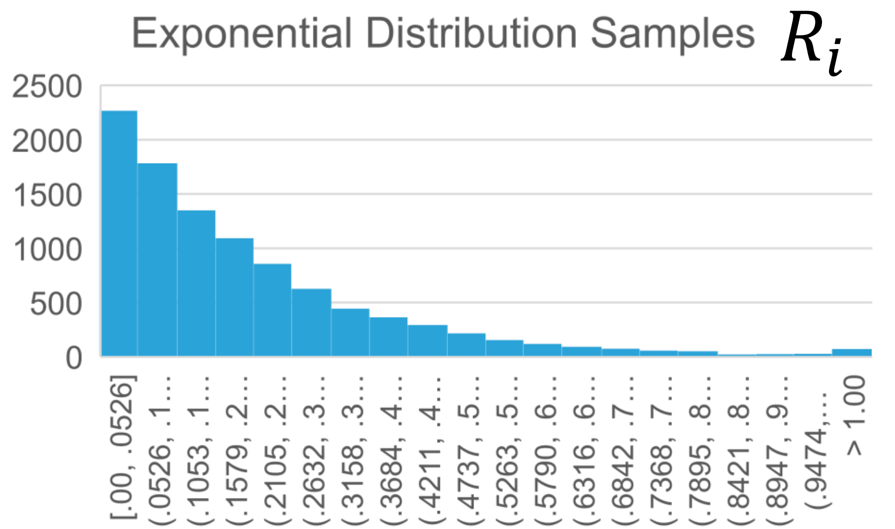
Warping Uniform To Exponential Distribution

```
def warp_expx(X, lambda: float):
    return –np.log(1.0 – X) / lambda

LAMBDA = 0.5

samples_uniform = np.random.rand(N)
samples_exp_ref = np.random.exponential(1.0/LAMBDA, N)
samples_exp_gen = warp_expx(samples_uniform, LAMBDA)

h1, h2, h3 = histograms(0.0, 1.0, 20, samples_uniform, samples_exp_ref, samples_exp_gen)

show_histogram(h1)
show_histogram(h2)
show_histogram(h3)
```

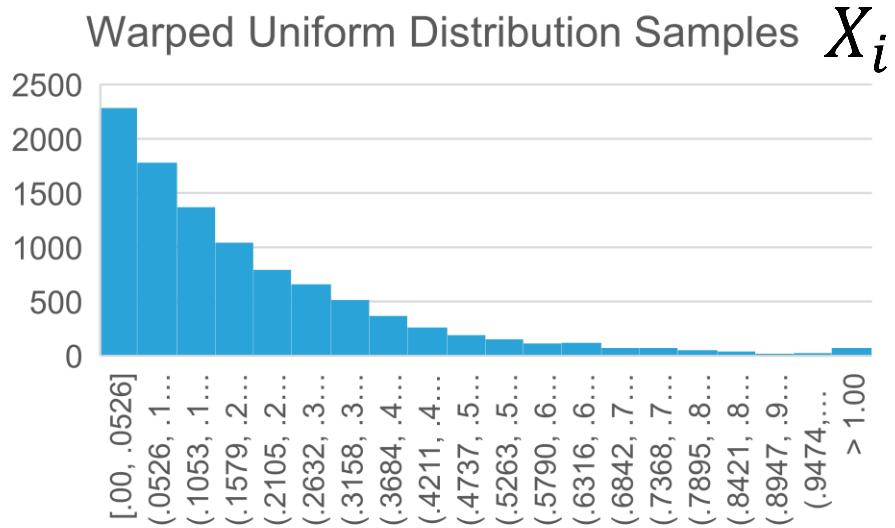
Warping Uniform To Exponential Distribution

- Histograms of generated samples





$$X_i = P_X^{-1}(\xi_i)$$



Mix Multiple Random Variables

- Let’s add another variable and combine them for 2D output

- In doing so, we are extending our sampling *domain*

- The sampling domain is defined by
  - The number of variables, and
  - Their respective ranges

- Think of the domain as a space with the axes representing variables

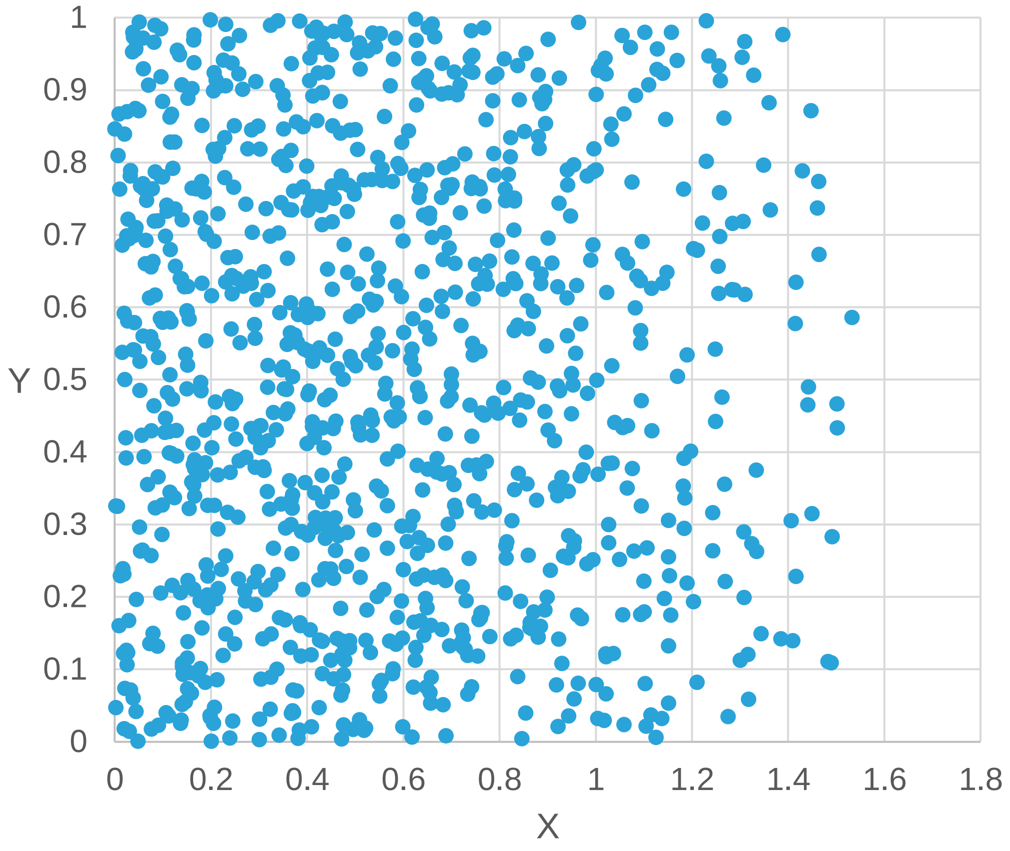
Joint PDF

- If multiple variables are in a domain, the **joint** PDF probability density of a given point in that domain depends on all of them
- In the simplest case, with independent variables X and Y, the joint PDF of their shared domain PDF is simply $p(x, y) = p_X(x)p_Y(y)$
- We can sample independent variables in a domain by computing and sampling the inverse of their respective CDFs, separately

Inversion Method Examples in 2D

- 2D with $Y = \xi$. For X, use $X \in [0, \frac{\pi}{2})$ and $p(x) = \cos x$
- $P_X(x) = \int p(x)\, dx = \int \cos x\, dx = \sin x$
- $P_X^{-1}(\xi) = \sin^{-1}(\xi)$

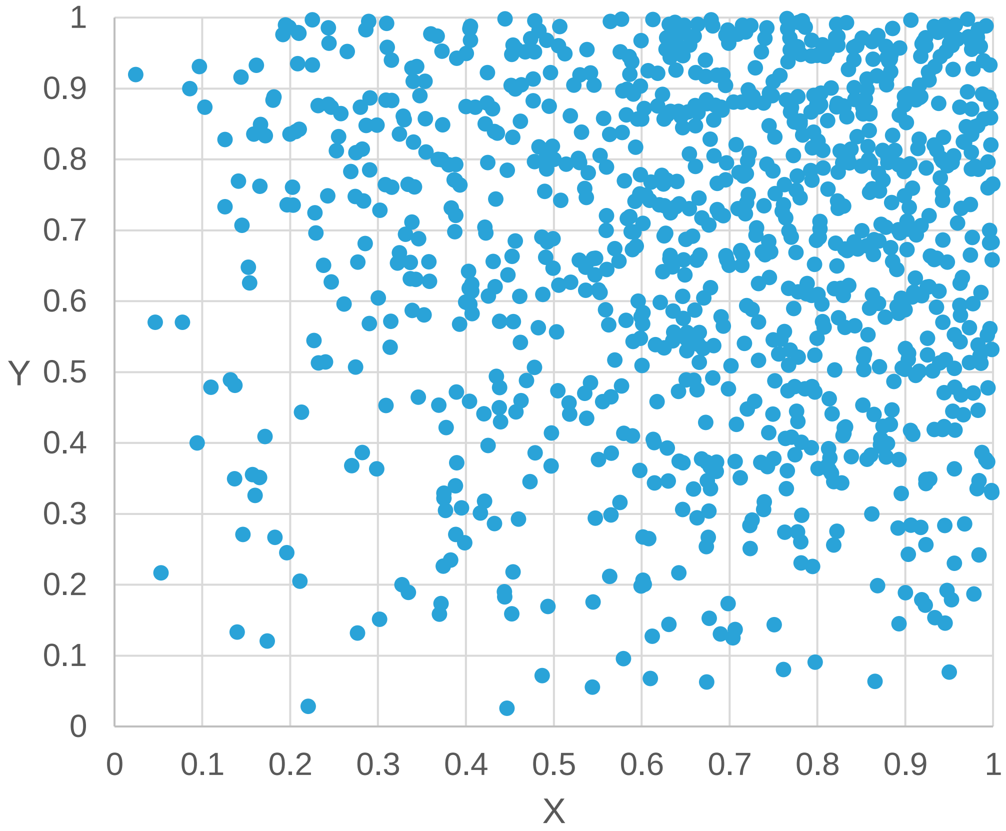
```
def gen_cosx(a: float, b: float, N: int):
    xi = np.random.rand(N)
    return np.arcsin(xi)


def p_cosx(x, a: float, b: float):
    return np.cos(x)


X_i = gen_cosx(0, 1, 1000)
plot(X_i, np.random.rand(1000))
```



Inversion Method Examples in 2D

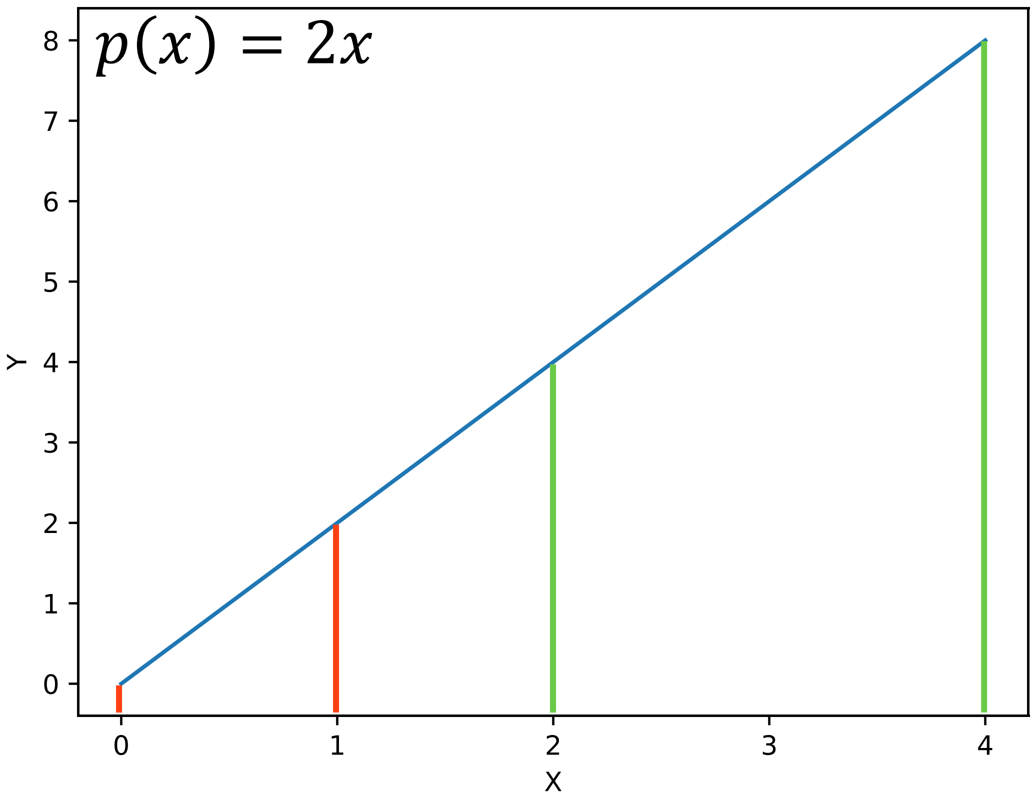
- X and Y in range $[0,1)$
- For both variables, $p(v) = 2v, P(v) = v^2, P^{-1}(\xi) = \sqrt{\xi}$

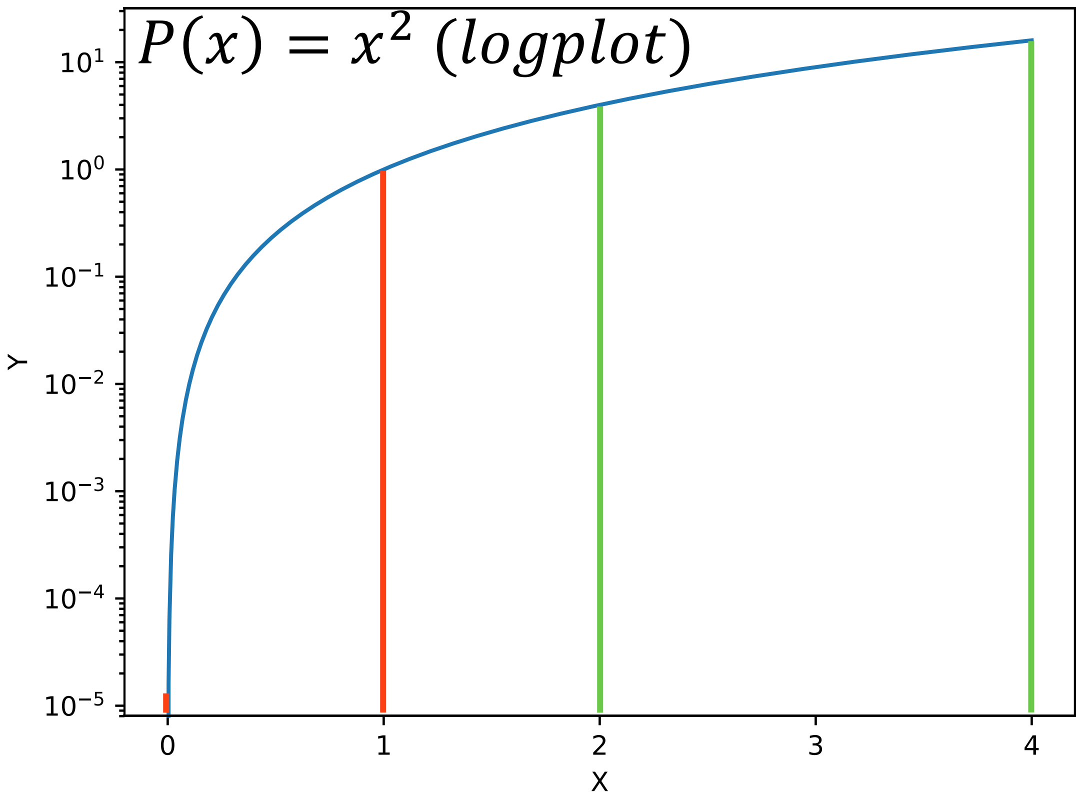
```
def gen_2v(a: float, b: float, N: int):
    xi = np.random.rand(N)
    return np.sqrt(xi)

def p_2v(v, a: float, b: float):
    return 2*v

X_i = gen_2v(0, 1, 1000)
Y_i = gen_2v(0, 1, 1000)
plot(X_i, Y_i)
```



Choosing a Different Range

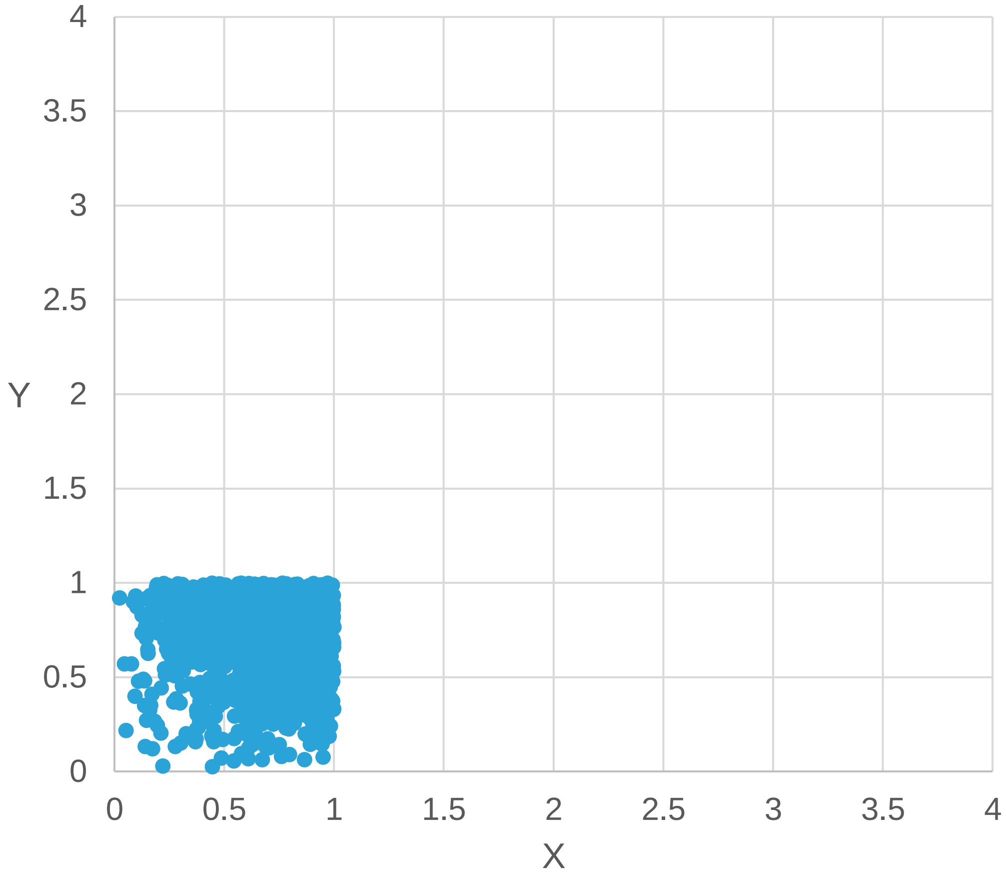
- Let's pick a slow-growing portion of the distribution function
- Compared to $[0,1)$, cumulative density only doubles in $[2,4)$





Inversion Method Examples in 2D

- Try X and Y in range $[2,4)$
- For both variables, $p(v) = 2v, P(v) = v^2, P^{-1}(\xi) = \sqrt{\xi}$
- Nothing happens.
- How can we adapt variable ranges?
- Something is missing!



Restricting the PDF / CDF

- Consider a given range from a to b for a variable and a candidate PDF $f(x)$ with the desired distribution shape

- If $\int_a^b f(x)\,dx \neq 1$, $f(x)$ is not a valid PDF for this variable

$$\Downarrow$$

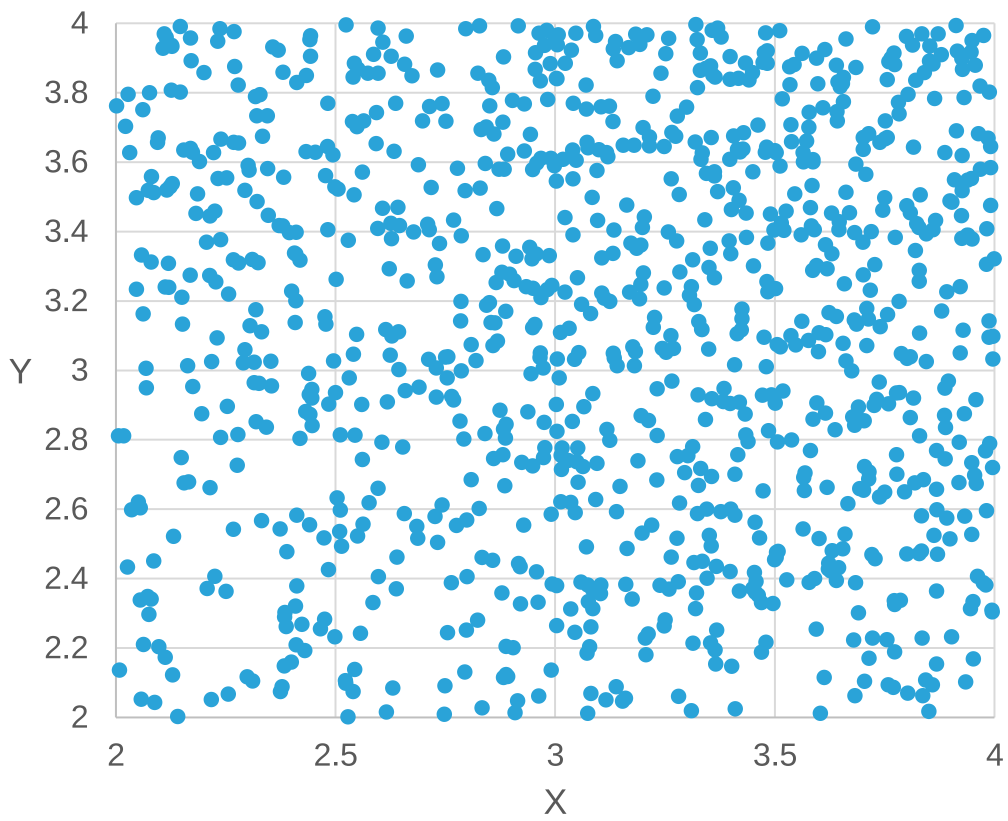
- <u>*The probability that the result is one of all possible results* $\neq 100\%$</u>

- To fix this, we compute the proportionality constant $c = \int_a^b f(x)\,dx$ and compute a valid $P(x) = \frac{F(x)}{c}$ and $p(x) = \frac{f(x)}{c} \propto f(x)$

Restricting the PDF / CDF

- For range $[a, b]$ where $a \neq 0$, we add a constant offset $k = -P(a)$

- Try $X, Y \in [2,4)$ and $f(v) = 2v$ again



- We compute $c_Y = c_X = \int_2^4 2v\, dv = 12$ and add $k = -\frac{4}{12}$ to get:

$$P(v) = \frac{v^2-4}{12},\ P^{-1}(\xi) = 2\sqrt{3 \cdot \xi + 1}, p(v) = \frac{2v}{12}$$

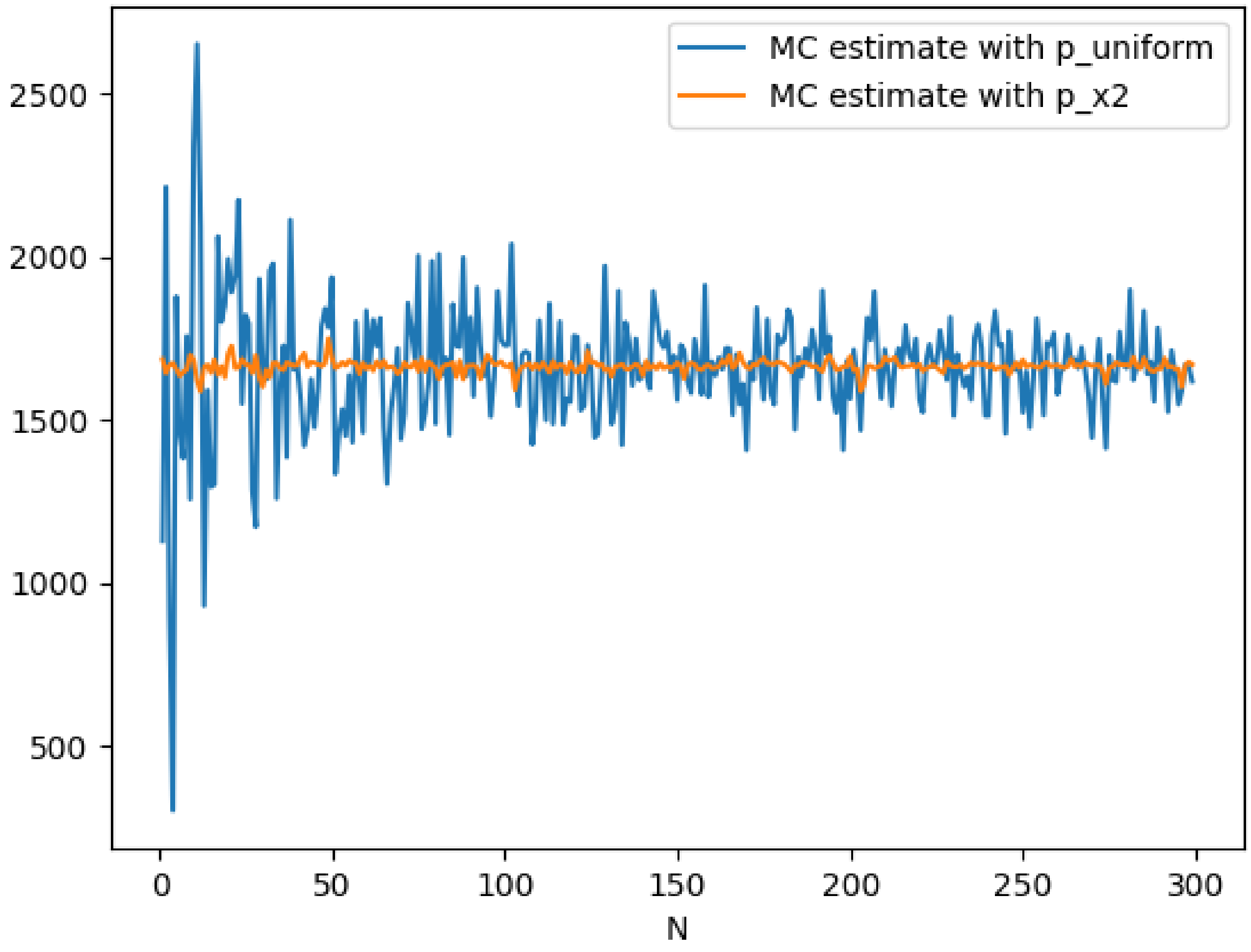
The Inversion Method, Completed

- Find a candidate function $f(x)$ with the desired distribution shape
- Choose the range $[a, b]$ in $f(x)$ you want your variable to imitate
- Determine the indefinite integral $F(x) = \int f(x)\, dx$
- Compute the proportionality constant $c = F(b) - F(a)$
- The CDF for the new variable X is $P_X(x) = \frac{F(x) - F(a)}{c}$
- Compute the inverse of the CDF $P_X^{-1}(\xi)$
- Use $P_X^{-1}(\xi)$ to warp the samples of a canonic random variable so that they are distributed with $p(x) = \frac{f(x)}{c}$ in the range $[a, b)$

Deriving the $p(x) \propto x^2$ Sample Generation Functions

integrate_mc(0, 100, N, f, p_uniform, gen_uniform) **vs** integrate_mc(0, 100, N, f, p_x2, gen_x2)

```
def integrate_mc(a: float, b: float, N: int, f, p, gen):
    X = gen(a, b, N)
    estimates = f(X)/p(X, a, b)
    result = estimates.sum() / N
    return result

def p_uniform(x, a: float, b: float):
    return x/(b-a)

def p_x2(x, a: float, b: float):
    b3 = ((b**3)/3)
    a3 = ((a**3)/3)
    return x**2/(b3-a3)
```



Deriving the $p(x) \propto x^2$ Sample Generation Functions

integrate_mc(0, 100, N, f, p_uniform, gen_uniform) **vs** integrate_mc(0, 100, N, f, p_x2, gen_x2)

```
def integrate_mc(a: float, b: float, N: int, f, p, gen):
    X = gen(a, b, N)
    estimates = f(X)/p(X, a, b)
    result = estimates.sum() / N
    return result

def p_uniform(x, a: float, b: float):
    return x/(b-a)

def p_x2(x, a: float, b: float):
    b3 = ((b**3)/3)
    a3 = ((a**3)/3)
    return x**2/(b3-a3)
```

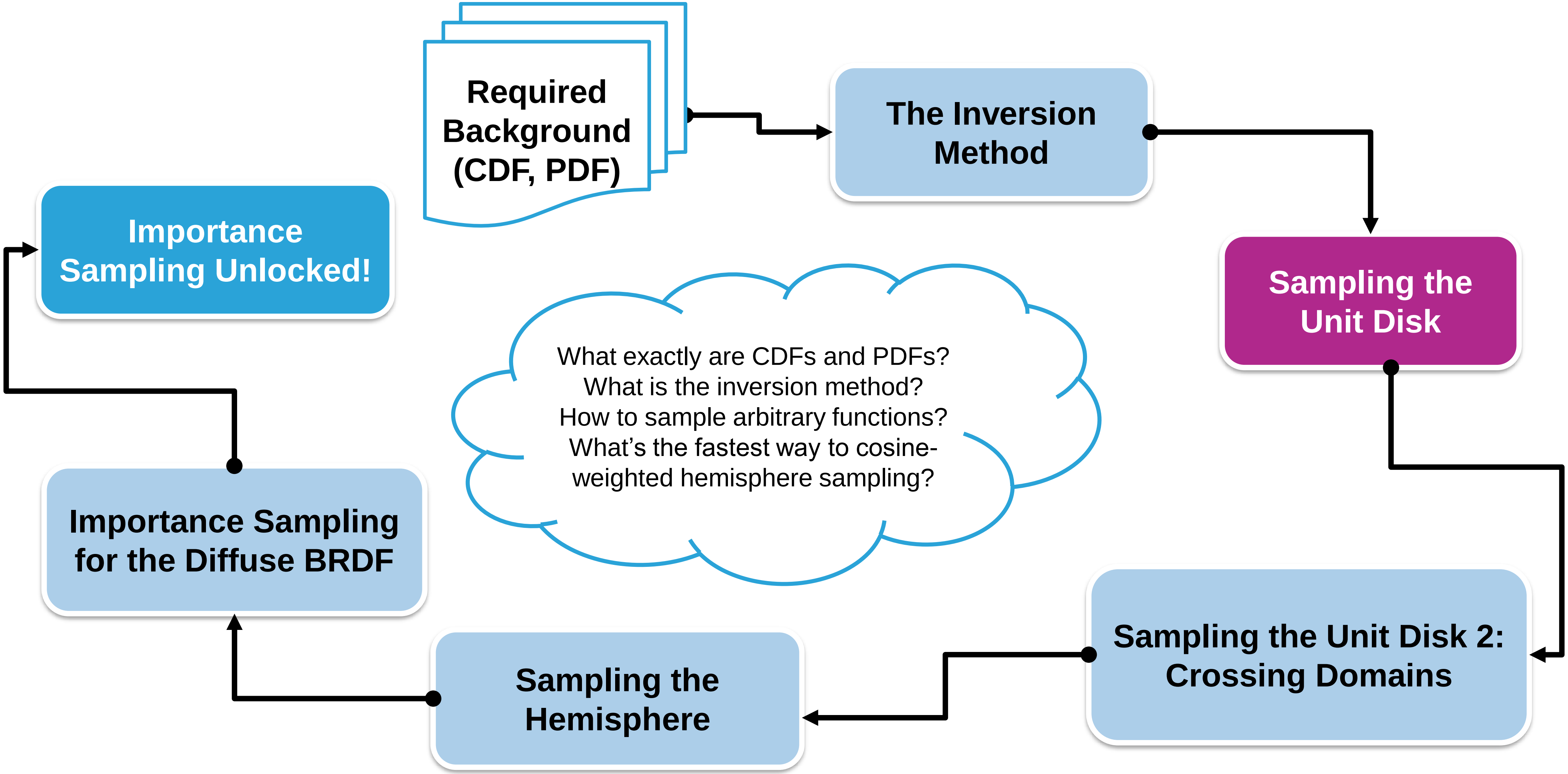
$$F(x) = \frac{x^3}{3}, c = \frac{b^3 - a^3}{3},$$

$$p(x) = \frac{x^2}{c},$$

$$P_X^{-1}(\xi) = \sqrt[3]{a^3 + \xi(b^3 - a^3)},$$

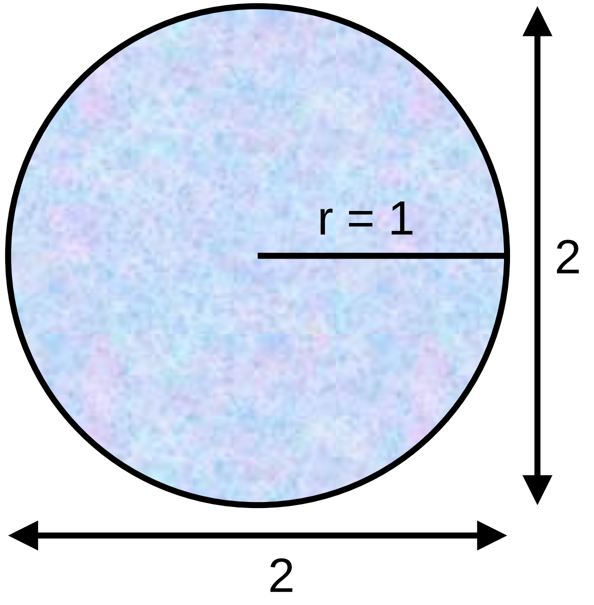
```
def gen_uniform(a: float, b: float, N: int):
    xi = np.random.rand(N)
    return xi * (b - a) + a

def gen_x2(a: float, b: float, N: int):
    xi = np.random.rand(N)
    b3 = (b**3)
    a3 = (a**3)
    return (a3+xi*(b3-a3))**(1.0/3.0)
```

Today’s Roadmap



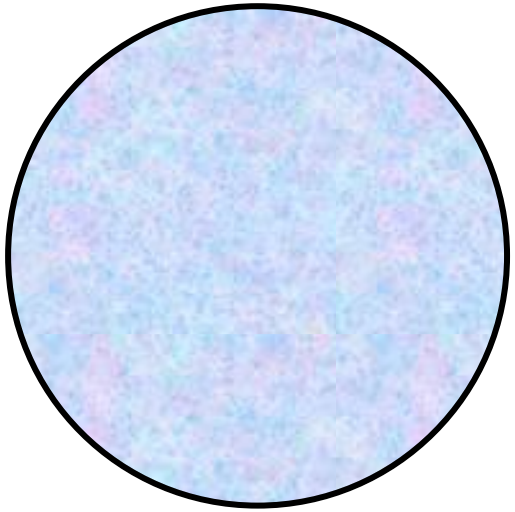
Sampling a Unit Disk

- Imagine we have a disk-shaped surface with radius $r = 1$ that registers incoming light (color) from directional light sources
- As an exercise, we want to approximate the total incoming light over the disk's surface **area**
- We integrate over an area of size π
- We will use the Monte Carlo integral for that



Uniformly Sampling the Unit Disk

- If we can manage to uniformly sample the disk, then we can compute the Monte Carlo integral as a simple average $\times\ \pi$
- By drawing uniform samples in x and y, we cannot cover the area precisely
- Inscribed square: information lost
- Circumscribed square: unnecessary samples



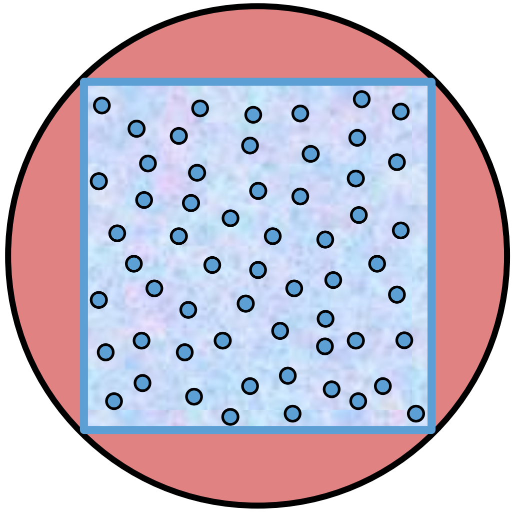
Uniformly Sampling the Unit Disk

- If we can manage to uniformly sample the disk, then we can compute the Monte Carlo integral as a simple average

- By drawing uniform samples in x and y, we cannot cover the area precisely

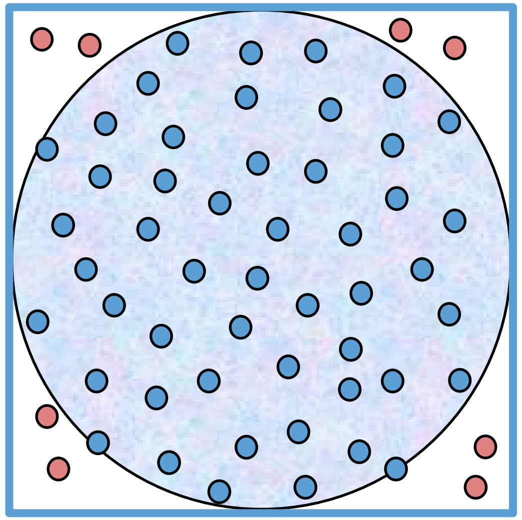
- Inscribed square: information lost

- Circumscribed square: unnecessary samples



Uniformly Sampling the Unit Disk

- If we can manage to uniformly sample the disk, then we can compute the Monte Carlo integral as a simple average
- By drawing uniform samples in x and y, we cannot cover the area precisely
- Inscribed square: information lost
- Circumscribed square: unnecessary samples



Back to the Unit Disk

- We do not want to waste samples if we can avoid it
- Instead, find a way to generate uniform samples on the disk
- Create samples in a non-cartesian domain: 2D polar coordinates
  - Polar coordinates defined by radius $r \in [0,1)$ and angle $\theta \in [0,2\pi)$
  - Transformation to cartesian coordinates:
    $x = r \sin\theta$
    $\mathrm{y} = r \cos\theta$

Uniformly Sampling the Unit Disk?

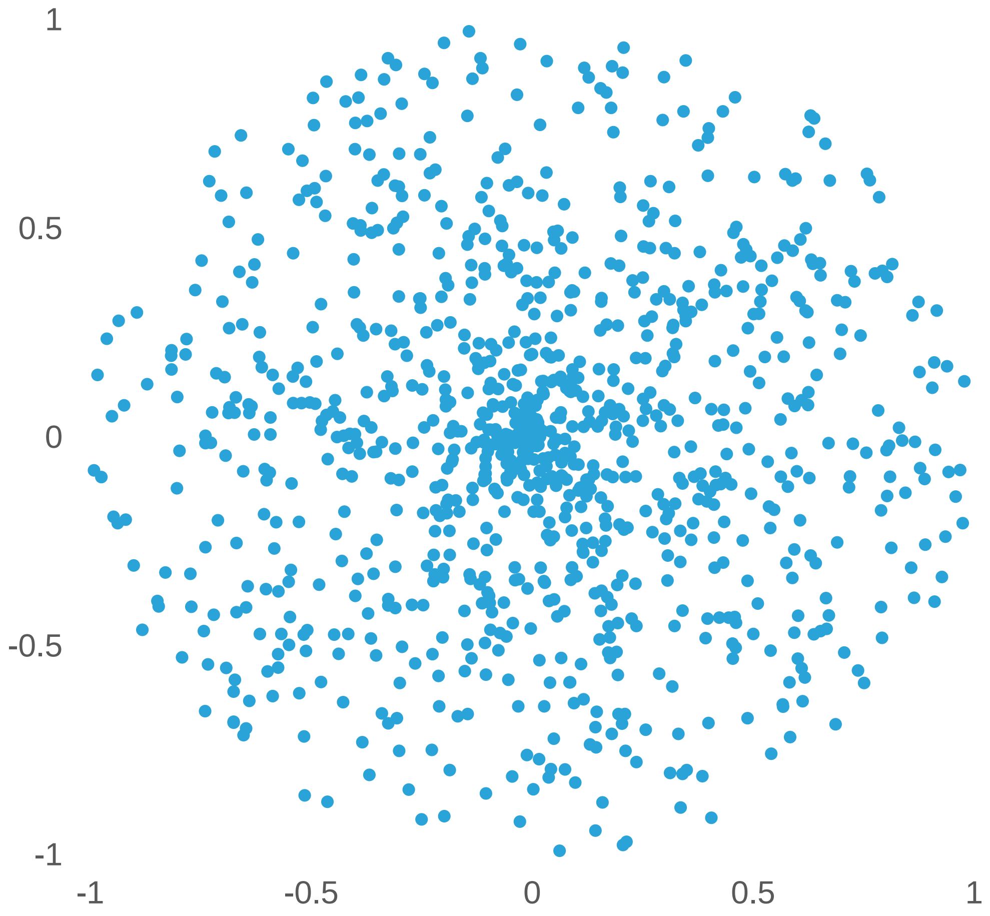
- Convert two ξ to ranges $[0,1), [0,2\pi)$ to get polar coordinates

- Convert to cartesian coordinates

```
void sampleUnitDisk()
{
	std::default_random_engine r_rand_eng(0xdecaf);
	std::default_random_engine theta_rand_eng(0xcaffe);

	std::uniform_real_distribution<double> uniform_dist(0.0, 1.0);

	for (int i = 0; i < NUM_SAMPLES; i++)
	{
		auto r = uniform_dist(r_rand_eng);
		auto theta = uniform_dist(theta_rand_eng) * 2 * M_PI;
		auto x = r * sin(theta);
		auto y = r * cos(theta);

		samples2D[i] = std::make_pair(x, y);
	}
}
```

Clumping

- We successfully sampled the unit disk in the proper range
- However, the distribution is not uniform with respect to the area
- Samples clump together at center
- Averaging those samples will give us a skewed result for the integral!

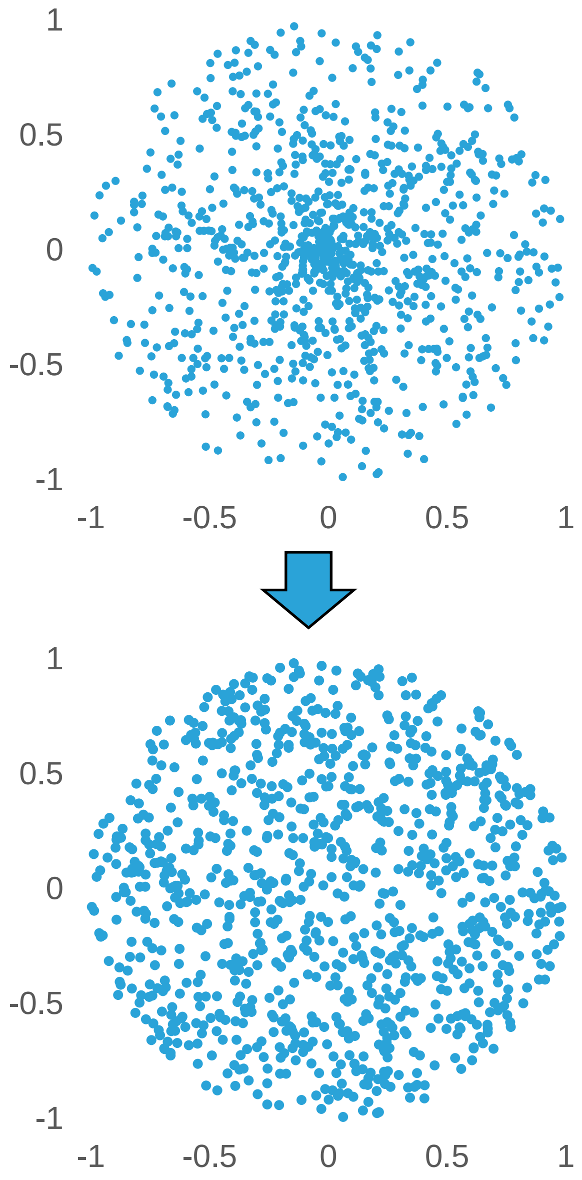


Uniformly Sampling the Unit Disk: A Solution

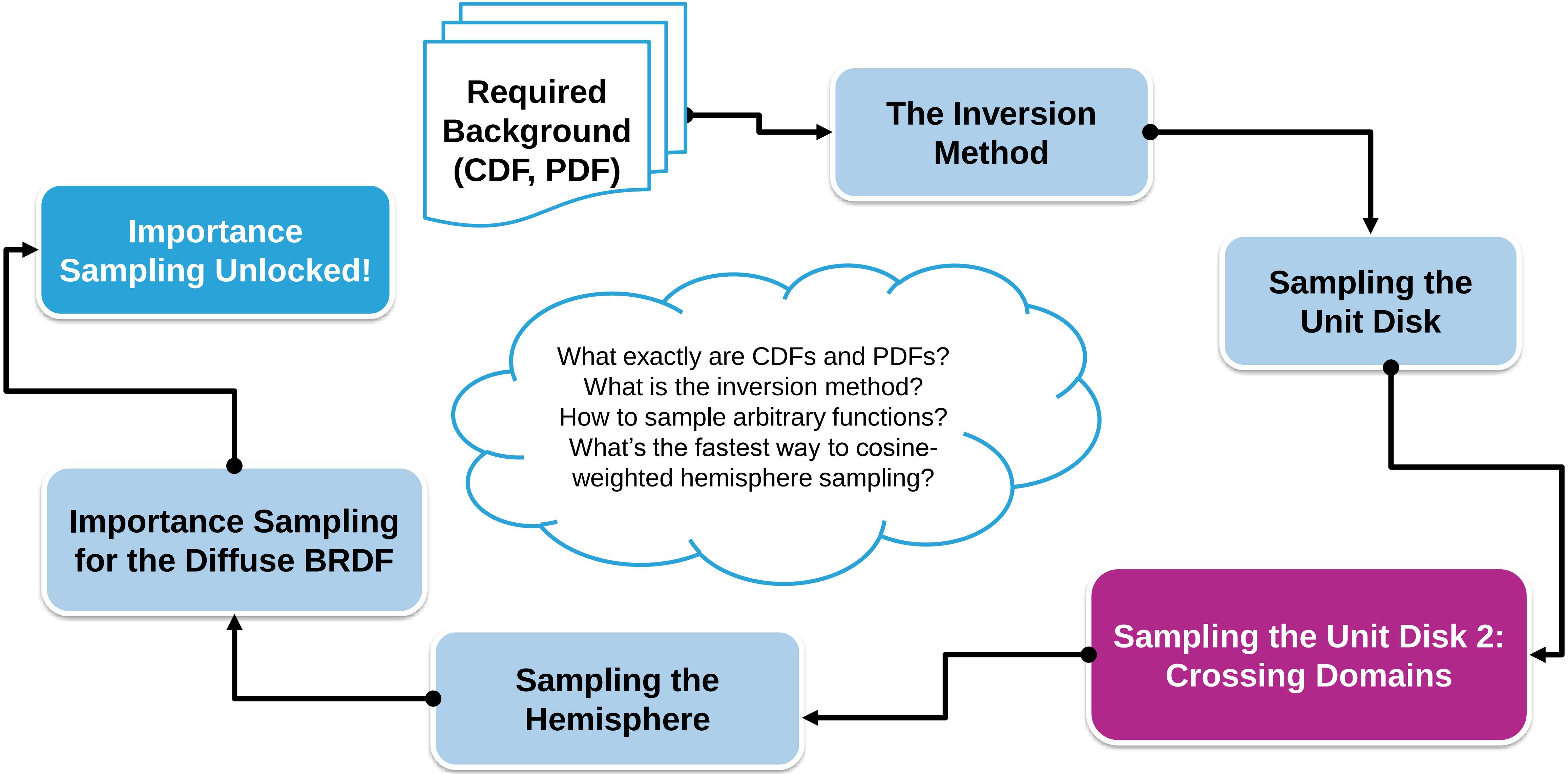
- The area of a disk is proportional to r^2, times a constant factor π
- If we see the disk as concentric rings of width Δr, the j inner rings up to radius $r_j = j\Delta r$ should contain $\left(\frac{r_j}{r}\right)^2 N$ out of N total samples
- Conversely, the i^{th} sample should lie in the ring at radius $r_i = r\sqrt{\frac{i}{N}}$
- Since ξ is uniform in $[0, 1)$, we can switch $\frac{i}{N}$ for ξ_i to get $r_i = r\sqrt{\xi_i}$

Uniformly Sampling the Unit Disk: A Solution

- It works, and it is not even a bad way to arrive at the correct solution
- However, for more complex scenarios, we might struggle to find the solution so easily
- With the tools we introduced earlier (and a few new tricks), we can formalize this process for arbitrary setups!



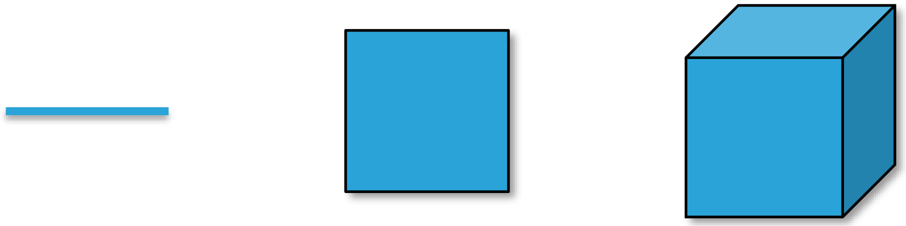
Today's Roadmap



Another Look at the PDF

- We saw samples being “warped”: we can interpret the inversion method as a spatial transformation of uniform samples

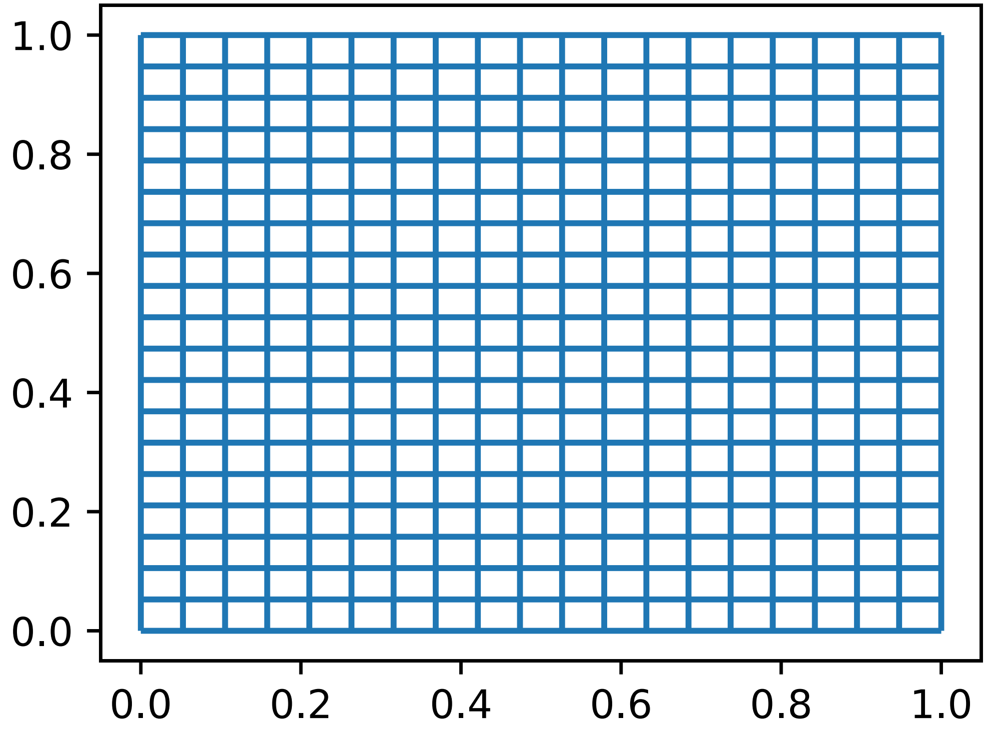
- Let’s treat the space in the input domain like a grid of infinitesimal *hypercubes*: segments in 1D, squares in 2D and cubes in 3D[5]



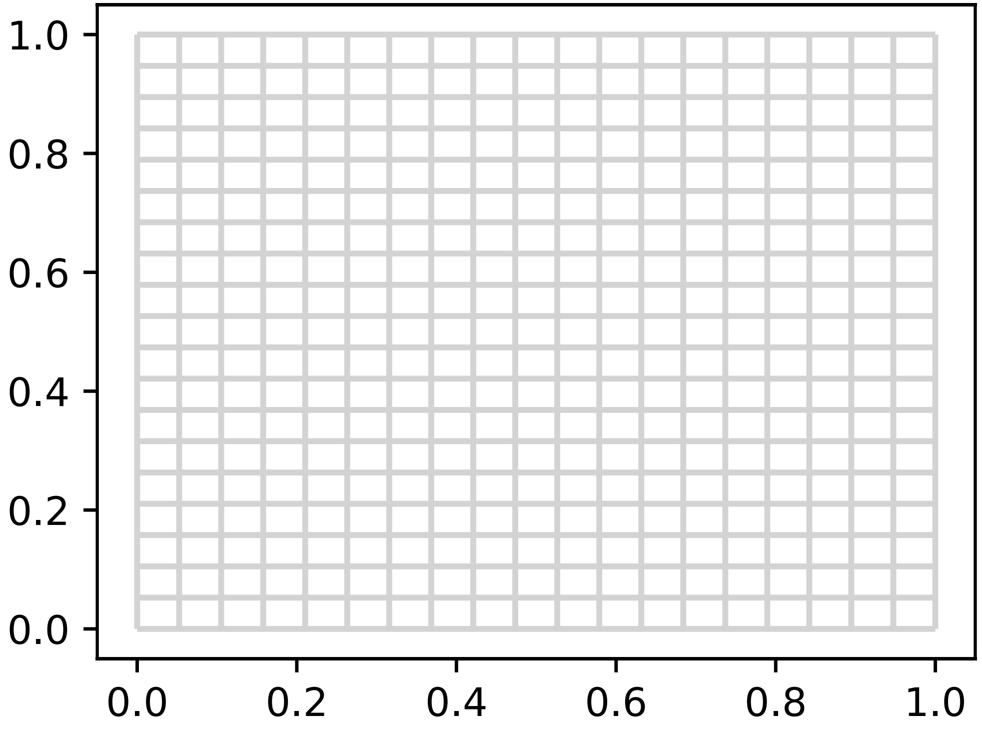
- If we warp a domain where each variable is ξ to one with joint PDF p_D, then $\frac{1}{p_D}$ is the change in volume of the hypercubes after warping

Visualizing the PDF in 2D

- The left represents our inputs and the right our target distribution
- This time, we warp **grid** coordinates with the inversion method



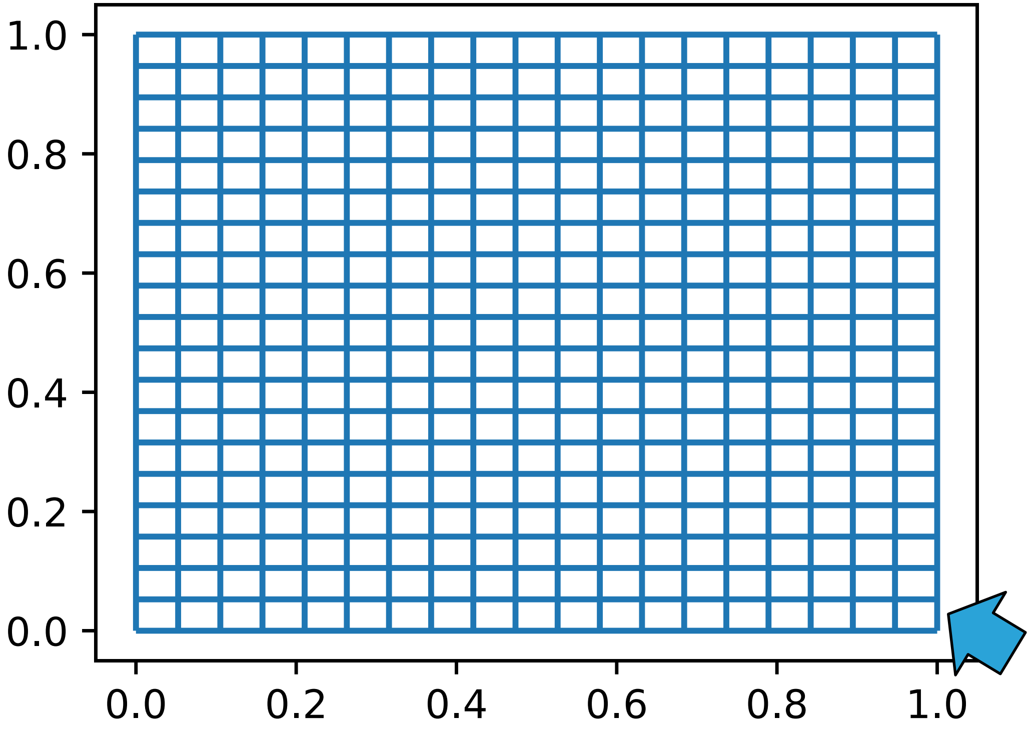
ξ_1, ξ_2



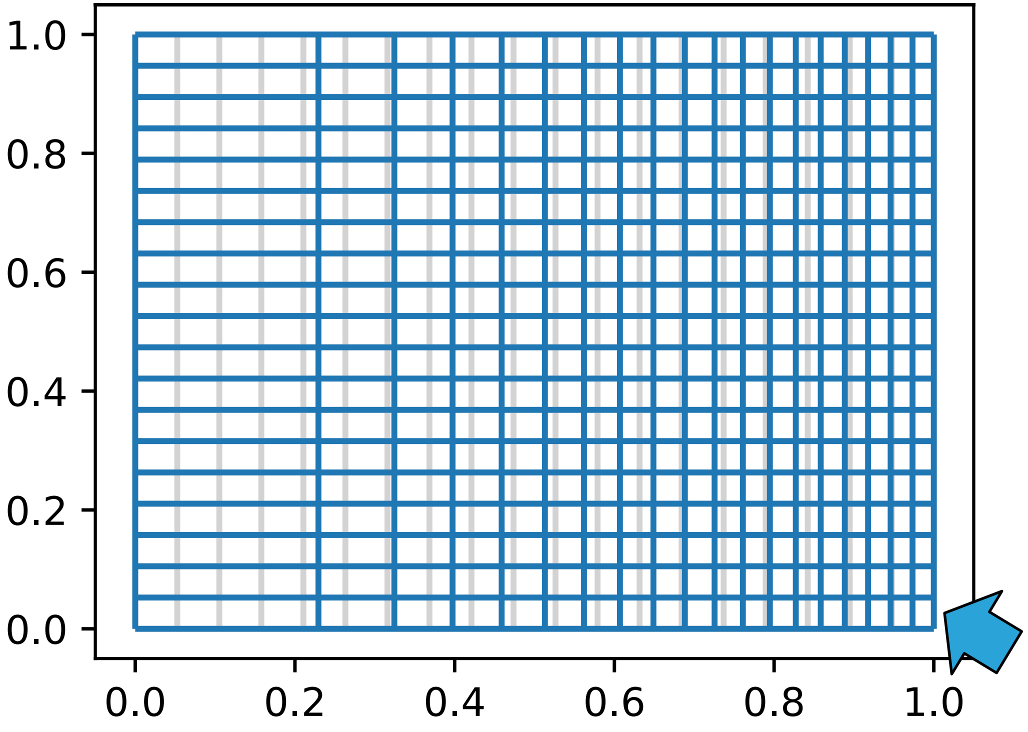
$Y = \xi_2$ and $X \in [0,1), p_X(x) = 2x$

Visualizing the PDF in 2D

- The areas of all 2D hypercubes (grid cells) are scaled by $\frac{1}{p_X(x)p_Y(y)}$
- $p_X(x) = 2x$, cells on the right at $(1, y)$ are half their original width



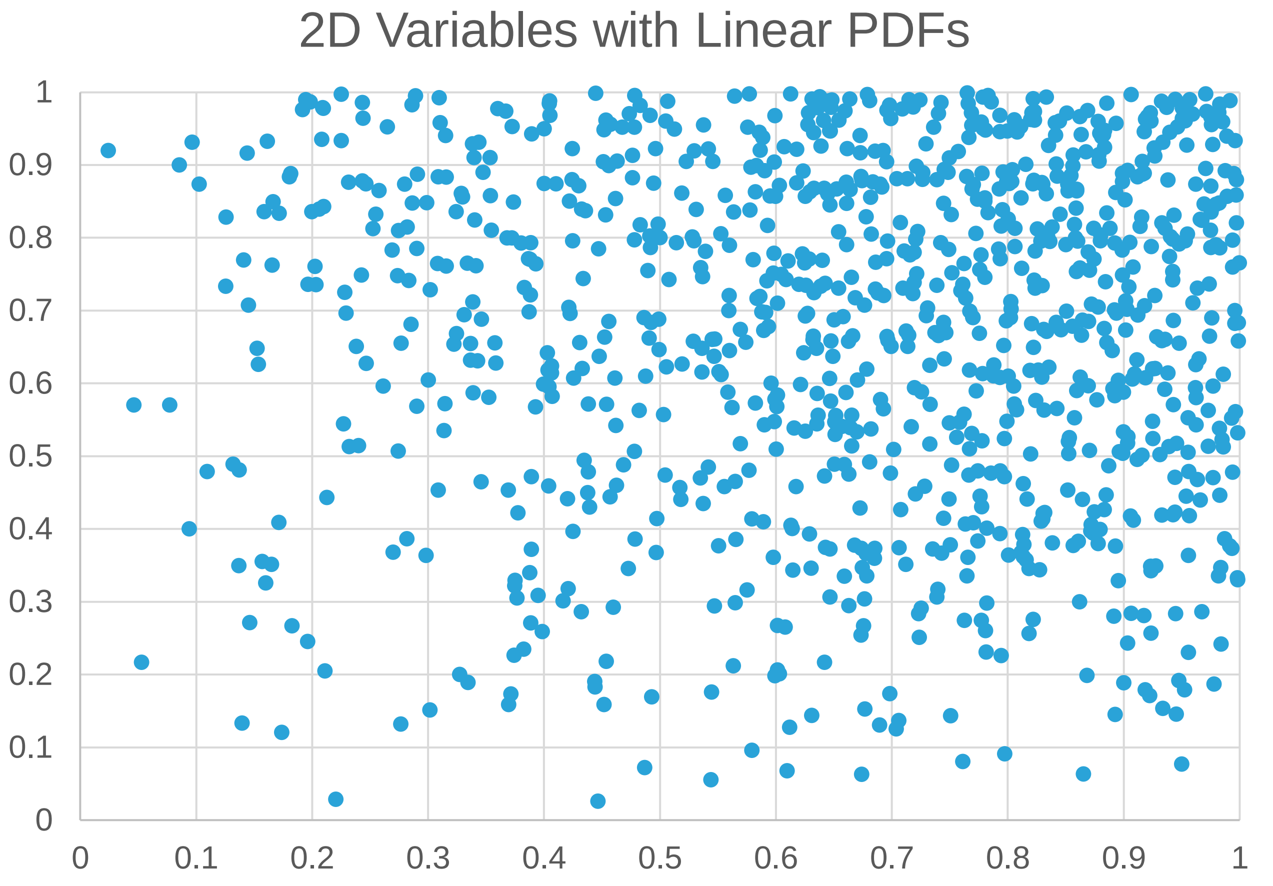
ξ_1, ξ_2



$Y = \xi_2$ and $X \in [0,1), p_X(x) = 2x$

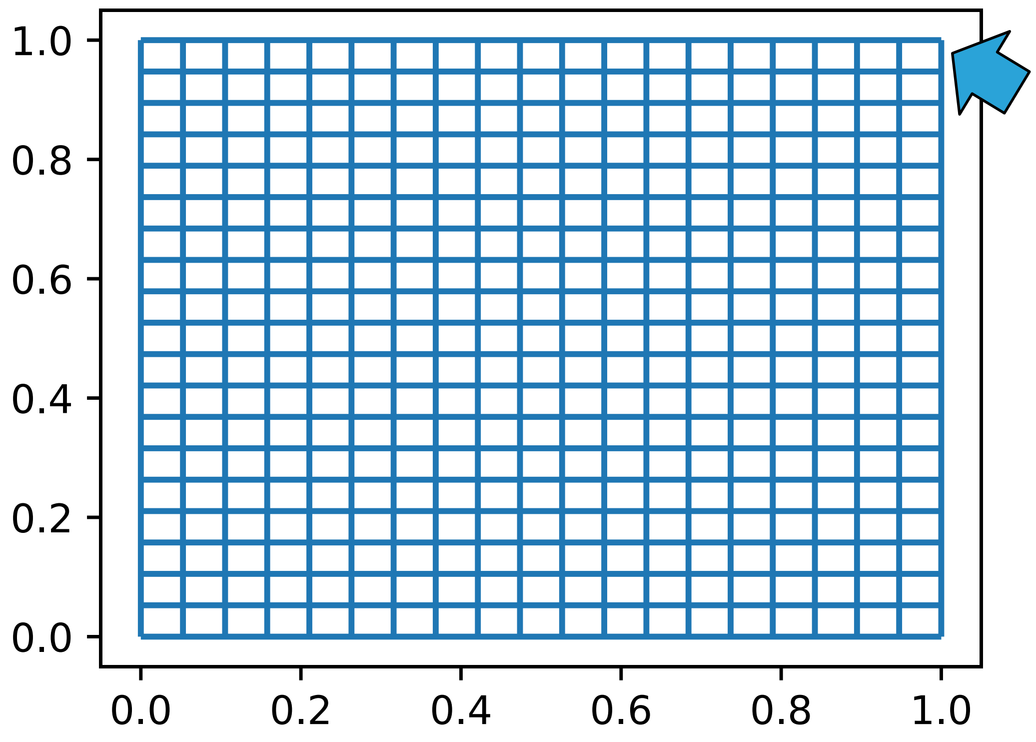
Visualizing the PDF in 2D

- Earlier, we saw samples $X, Y \in [0,1)$ with $p_X(x) = 2x, p_Y(y) = 2y$

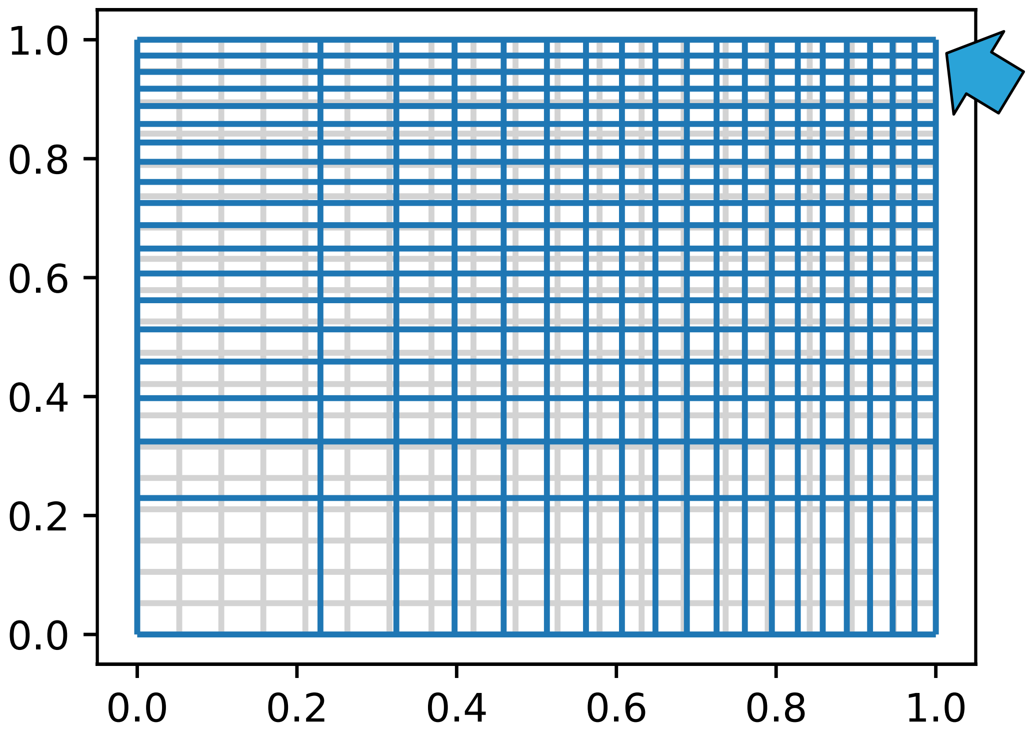


Visualizing the PDF in 2D

- In this 2D setup, we have joint PDF $p(x,y) = p_X(x)p_Y(y) = 4xy$
- Space near point $(1,1)$ is compressed down to $\frac{1}{4}$ of its original size



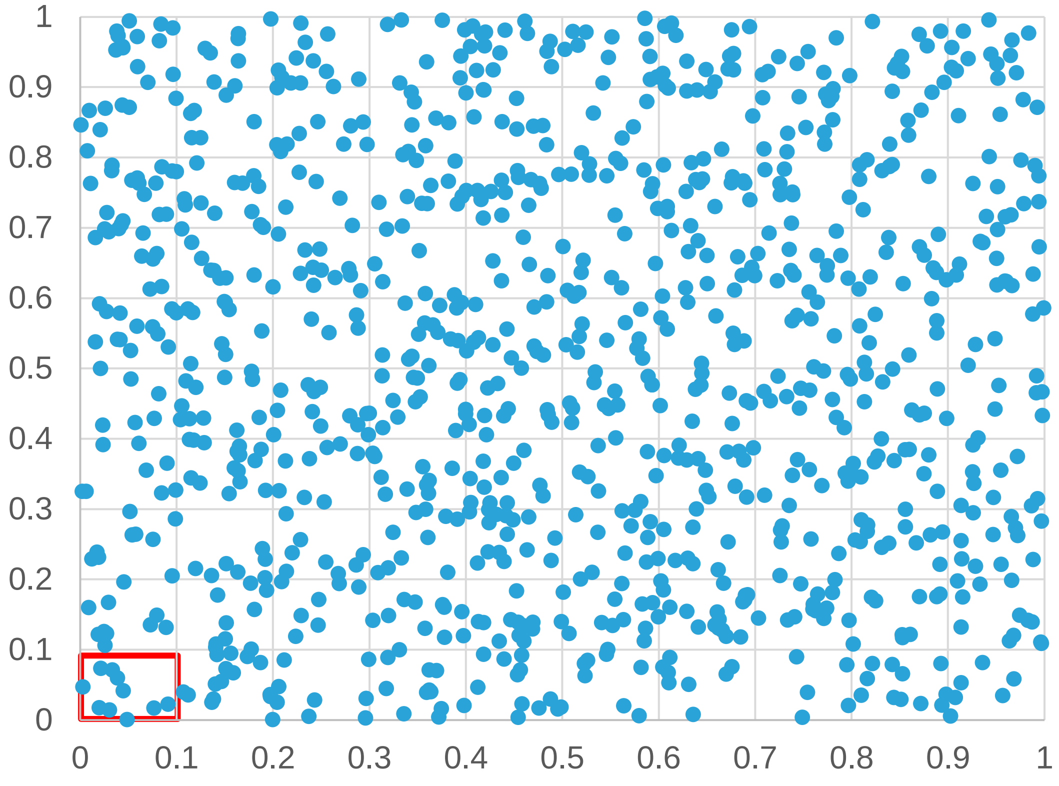
ξ_1, ξ_2



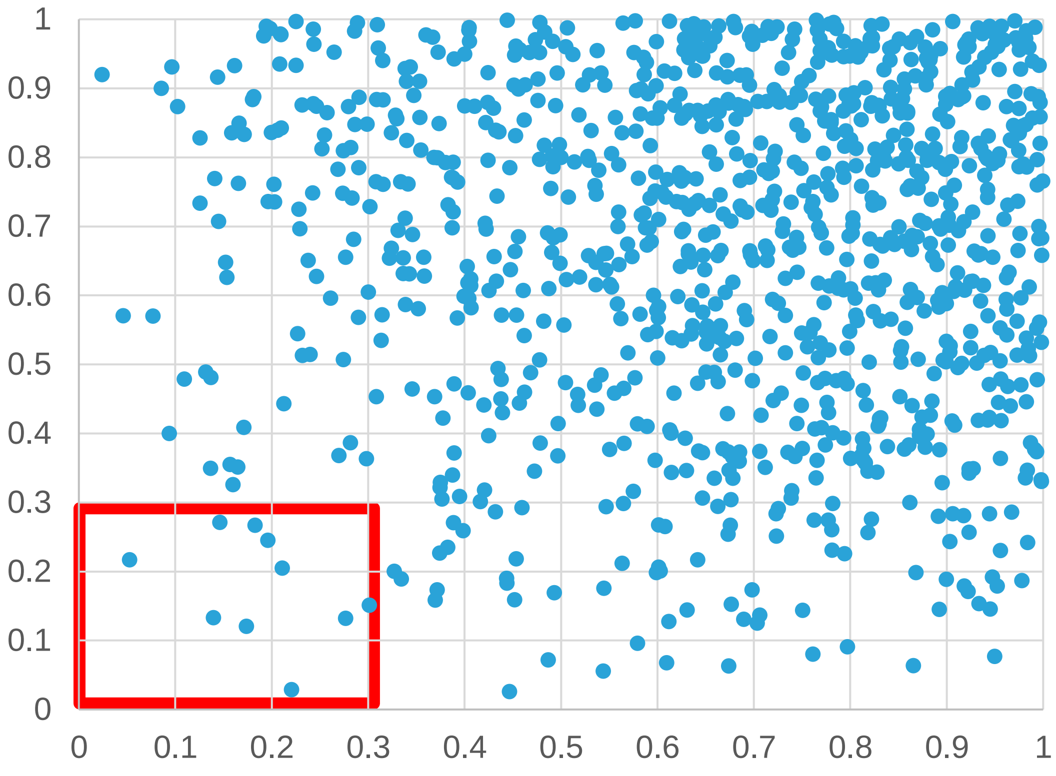
$X, Y \in [0,1), p_X(x) = 2x, p_Y(y) = 2y$

Visualizing the PDF in 2D

- This PDF compresses space at higher values of x, y, dilates at lower
- If space shrinks or grows, samples in it become denser or sparser



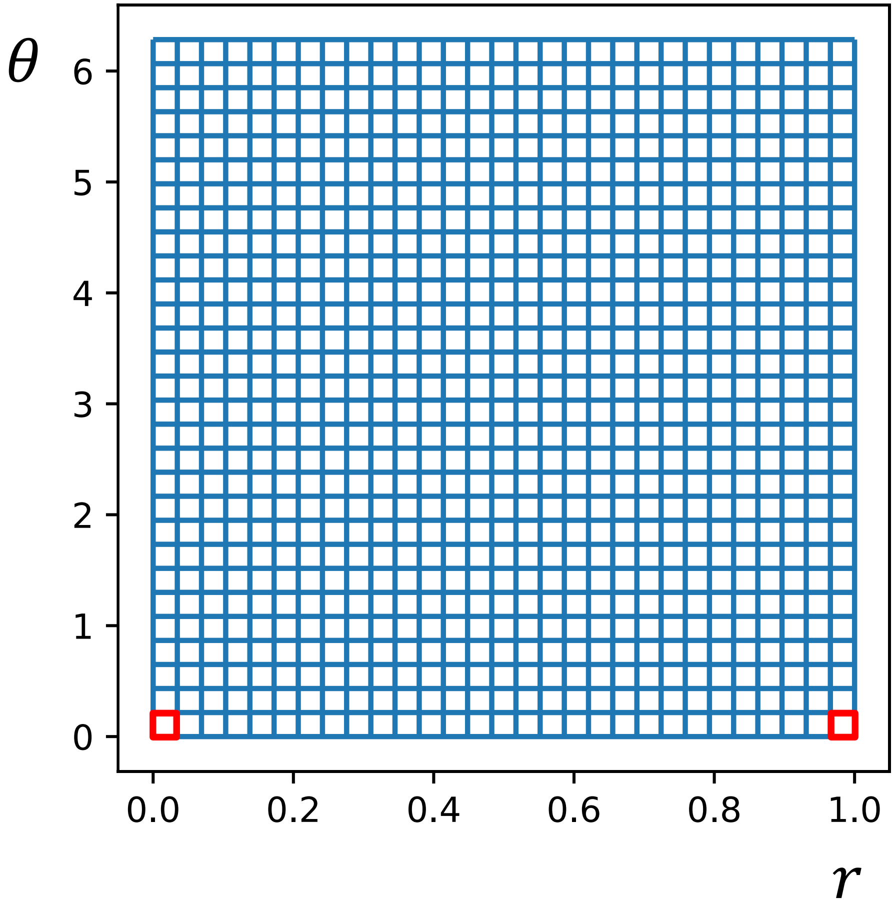
ξ_1, ξ_2



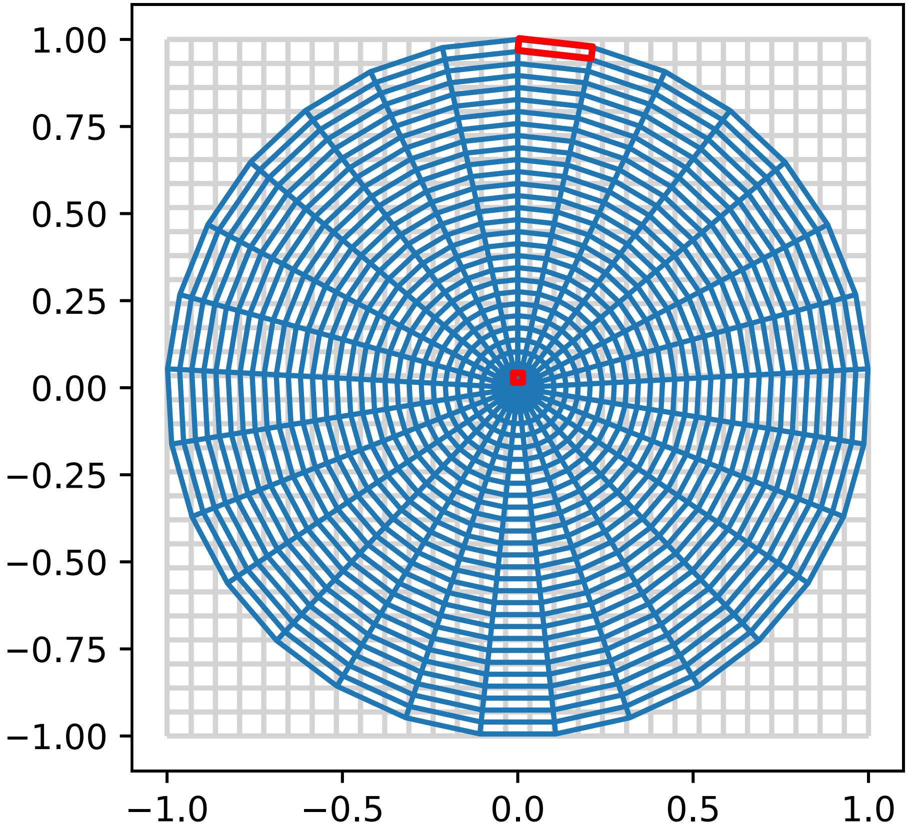
$X, Y \in [0,1), p_X(x) = 2x, p_Y(y) = 2y$

Polar To Cartesian Coordinates

- Let's transform a regular grid from polar to cartesian coordinates

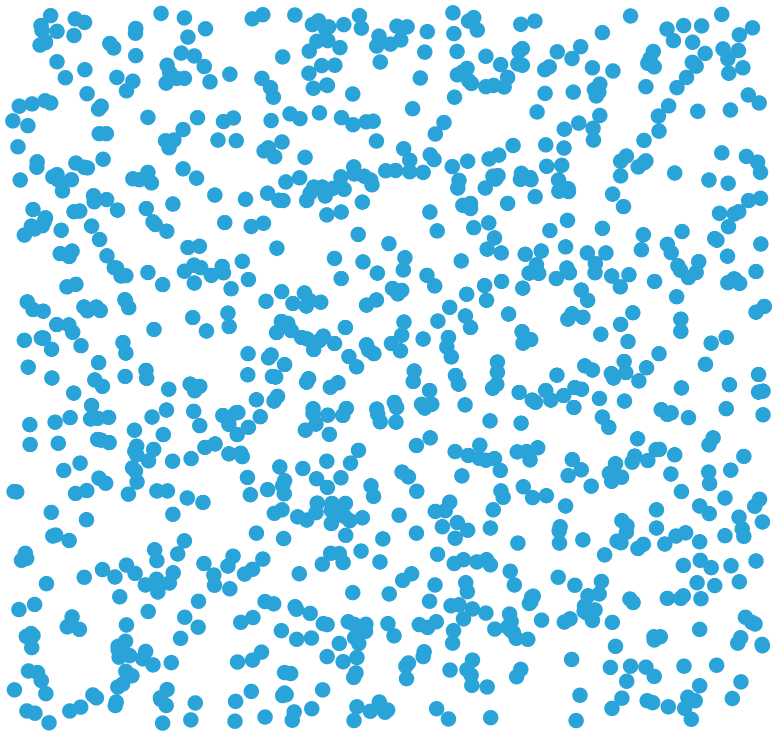


$$X = r\cos(\theta)$$
$$Y = r\sin(\theta)$$

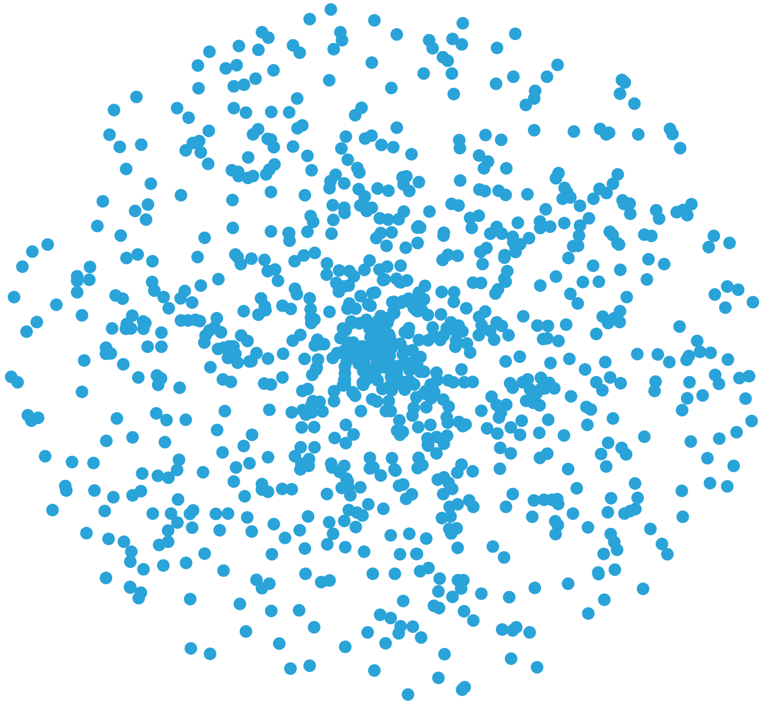


First Attempt to Learn the PDF

- Take 100k samples, transform and see in which square they end up

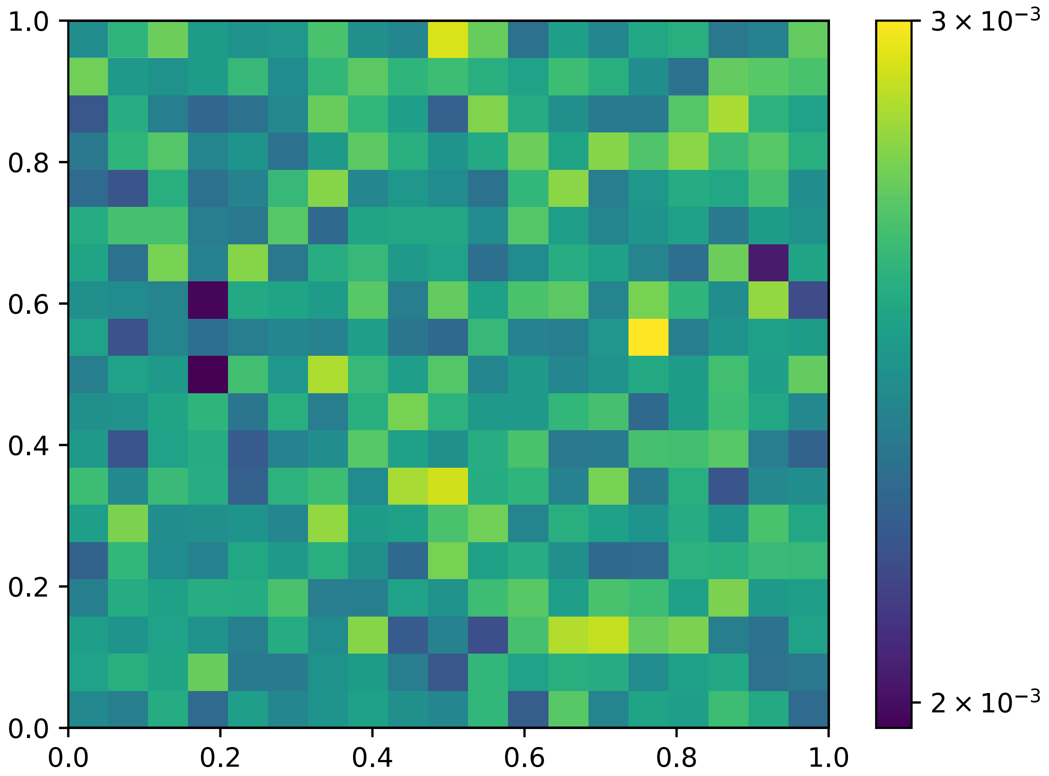


$$X = r\cos(\theta)$$
$$Y = r\sin(\theta)$$

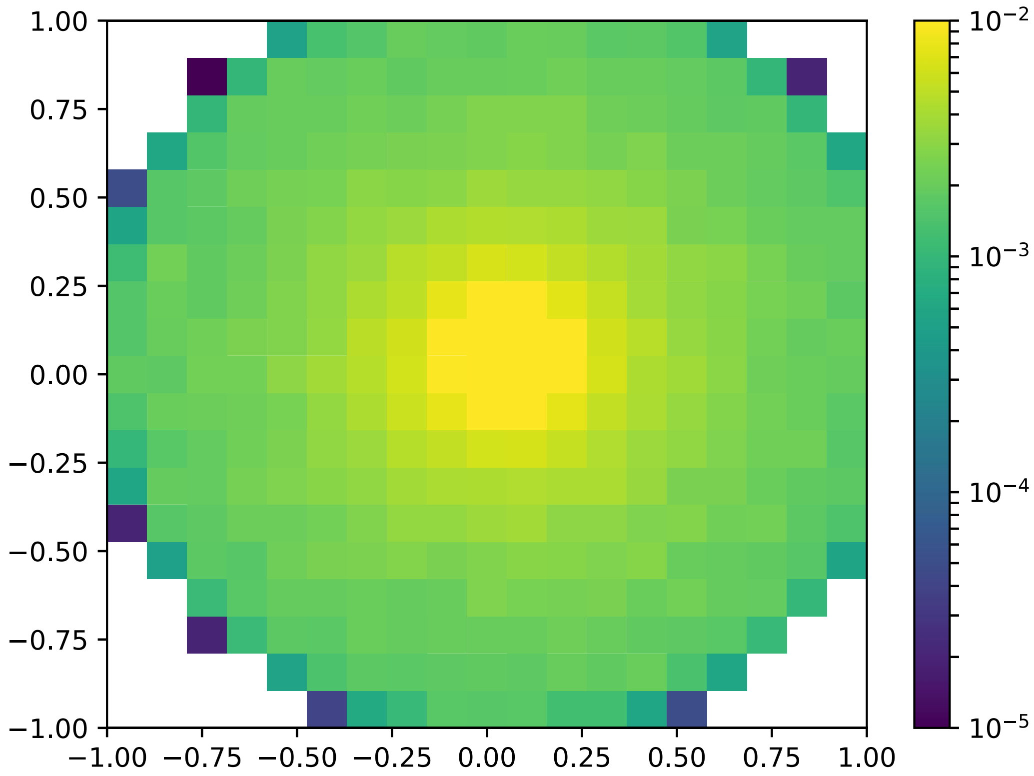


First Attempt to Learn the PDF

- Take 100k samples, transform and see in which square they end up



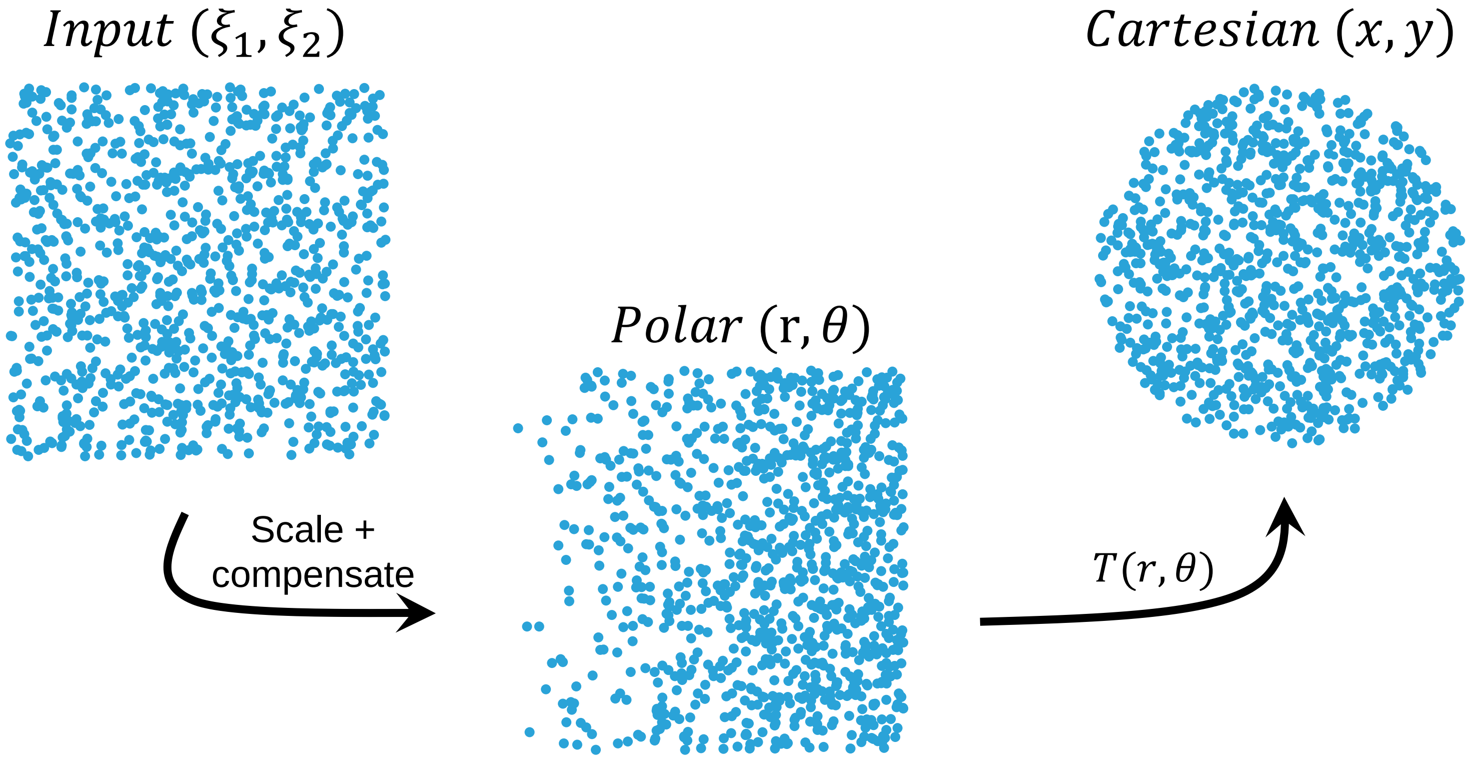
ξ_1, ξ_2



$X = \xi_1 \cos(2\pi\xi_2), Y = \xi_1 \sin(2\pi\xi_2)$

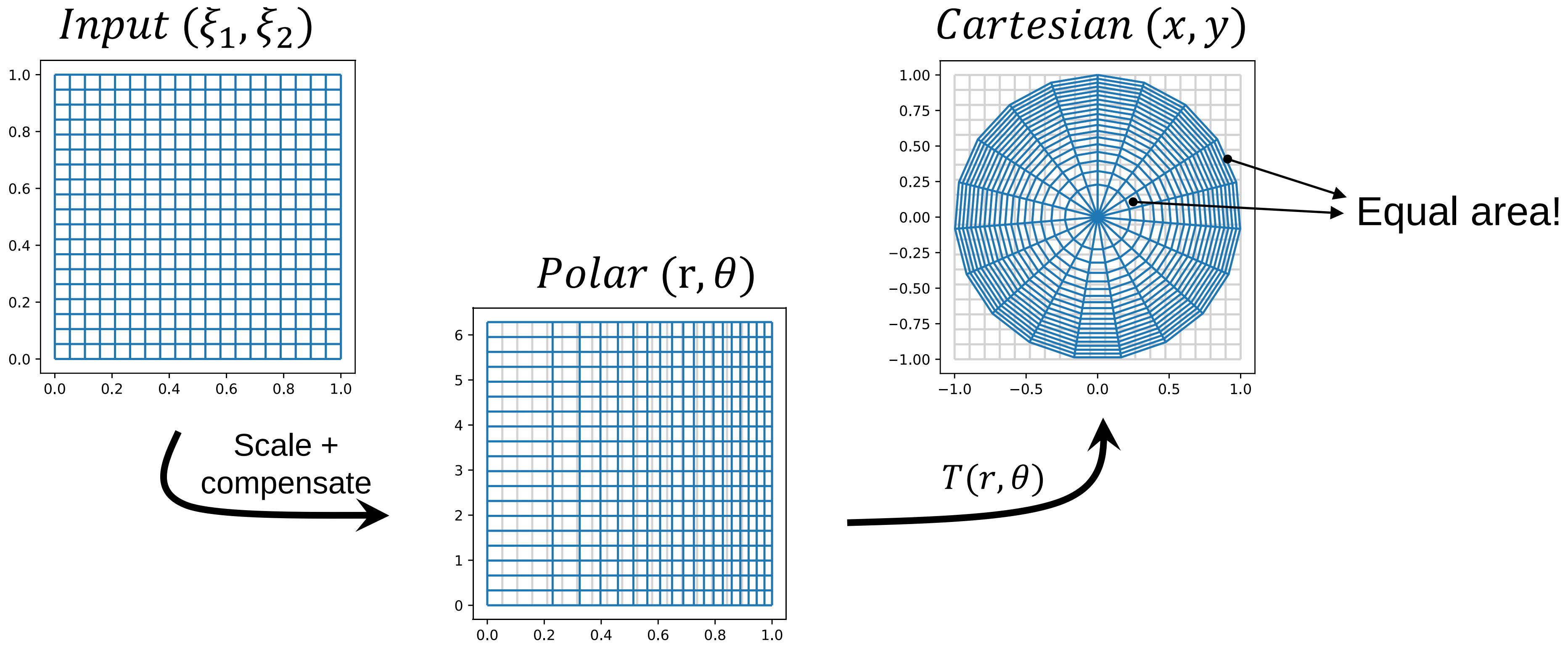
Knowing the PDF

- If we know the effect of a transformation T on the PDF, we can
  - Use it in the Monte Carlo integral to weight our samples, or
  - Compensate to get a uniform sampling method **after** transformation



Knowing the PDF

- If we know the effect of a transformation T on the PDF, we can
  - Use it in the Monte Carlo integral to weight our samples, or
  - Compensate to get a uniform sampling method **after** transformation



Computing the PDF after a Transformation

- Assume a random variable A and a **bijective** transformation T that yields another variable $B = T(A)$
- Bijectivity implies that $b = T(a)$ must be either monotonically increasing or decreasing with a
- This implies that there is a unique B_{i} for every A_{i}, and vice versa
- In this case, the CDFs for the two variables fulfill $P_B(T(a)) = P_A(a)$

Computing the PDF after a Transformation

- If $b = T(a)$ and b increases with a, we have: $\frac{dP_B(b)}{da} = \frac{dP_A(a)}{da}$
- If b decreases with a (e.g., $b = -a$), we have: $-\frac{dP_B(b)}{da} = \frac{dP_A(a)}{da}$

- Since p_B is the non-negative derivative of P_B, we can rewrite as:

$$p_B(b)\left|\frac{db}{da}\right| = p_A(a), \qquad Using: \frac{dP_X(x)}{dy} = \frac{p_X(x)\,dx}{dy}$$

$$p_B(b) = \left|\frac{db}{da}\right|^{-1} p_A(a)$$

Computing the PDF after a Transformation

- Let's interpret $p_B(b) = \left|\frac{db}{da}\right|^{-1} p_A(a)$
- It is the probability density of A at a, multiplied by $\left|\frac{db}{da}\right|^{-1}$
- $\left|\frac{db}{da}\right|^{-1}$ has two intuitive interpretations:

  the change in sample density at point a if we transform a by T
  or,
  the reciprocal change in volume (space) for a volume element (hypercube) at point a if we transform A by transformation T

Multidimensional Transformations

- If we try to apply the above to the unit disk, we fail at $x = r\sin\theta$
- We can't evaluate $\left|\frac{dx}{dr}\right|^{-1}$: the transformation that produces one target variable is dependent on both input variables and vice-versa
- We cannot compute the change in the PDF between individual variables, we must take them all into account simultaneously
- It's matrix time!

Multidimensional Transformations

- We write the set of N values from a **multidimensional** variable $\vec{A}$ as a vector $\vec{a}$ and the N outputs of transformation T as a vector $\vec{b}$:

$$\vec{a} = \begin{pmatrix} a_1 \\ \vdots \\ a_N \end{pmatrix}, \vec{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_N \end{pmatrix} = \begin{pmatrix} T_1(\vec{a}) \\ \vdots \\ T_N(\vec{a}) \end{pmatrix} = T(\vec{a})$$

- Instead of quantifying the change in volume incurred by $T(a)$, $\left|\frac{dT(a)}{da}\right|$, our goal is now to quantify the change incurred by $T(\vec{a})$

The Jacobian Matrix

- For a transformation $\vec{b} = T(\vec{a})$, we can define the Jacobian matrix that contains all b_j, a_i combinations of partial differentials

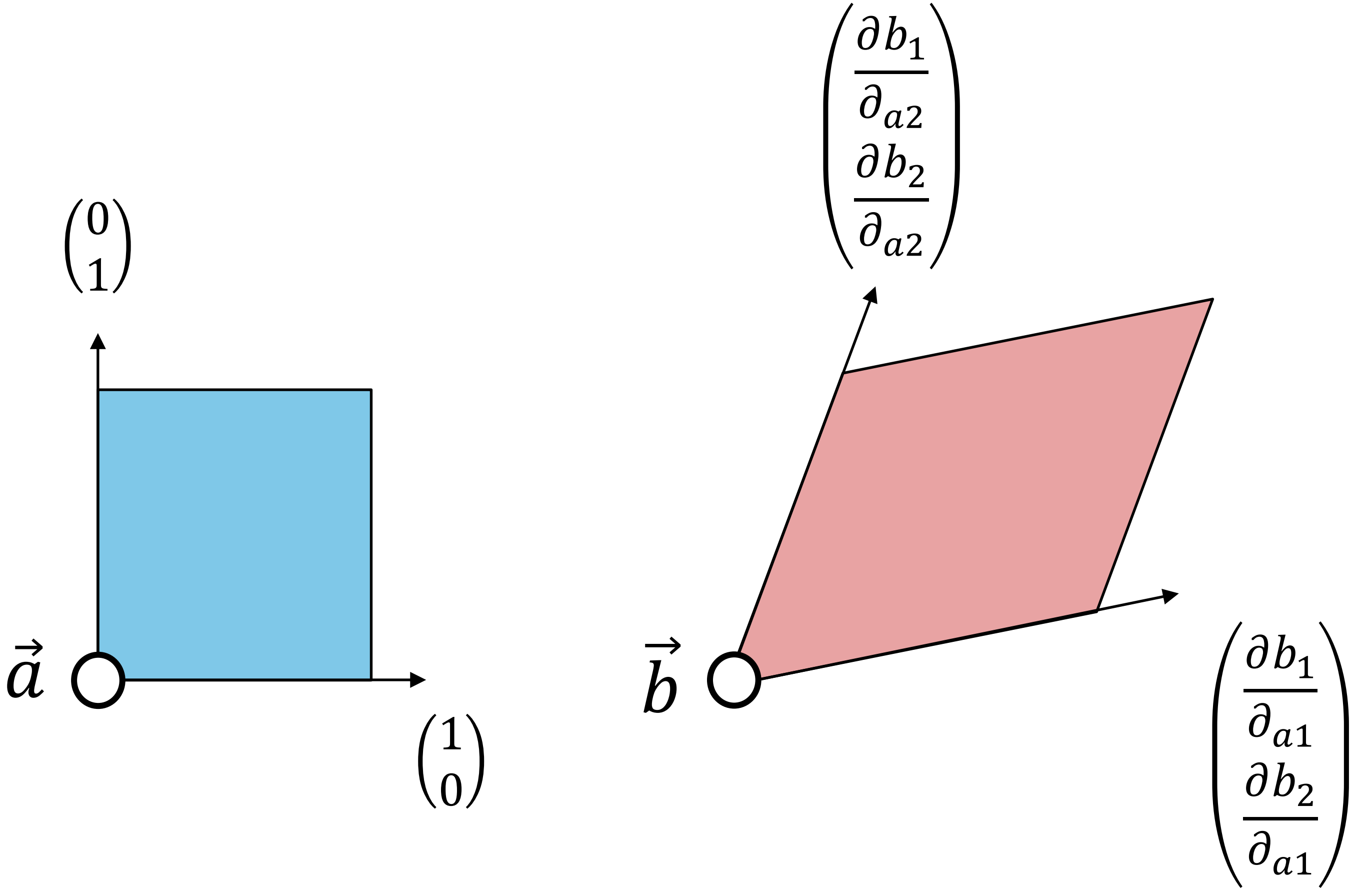
$$J_T(\vec{a}) = \begin{pmatrix} \frac{\partial b_1}{\partial a_1} & \cdots & \frac{\partial b_1}{\partial a_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial b_M}{\partial a_1} & \cdots & \frac{\partial b_M}{\partial a_N} \end{pmatrix}$$

- If we consider $\vec{A}$'s domain as a space with N axes, $J_T(\vec{a})$ gives the change of the edges of a volume element from $\vec{a}$ to $\vec{b} = T(\vec{a})$

The Jacobian Matrix, Visualized

- Change in edges of a volume element (infinitesimal hypercube) at $\vec{a}$

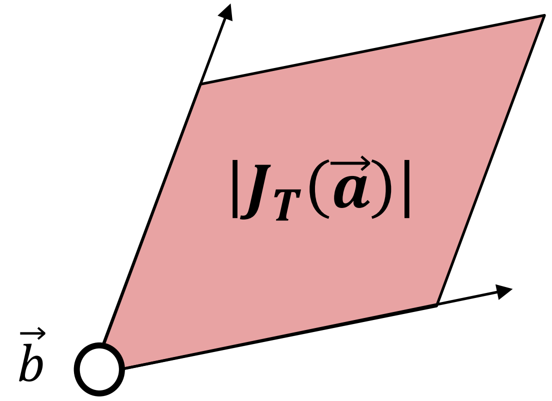
$$J_T(\vec{a}) = \begin{pmatrix} \frac{\partial b_1}{\partial a_1} & \cdots & \frac{\partial b_1}{\partial a_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial b_N}{\partial a_1} & \cdots & \frac{\partial b_N}{\partial a_N} \end{pmatrix}$$



The Jacobian

- The columns of a square matrix can be interpreted as the natural base vectors of a space $\begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}, \begin{pmatrix}0\\1\\\vdots\\0\end{pmatrix}$ if they were transformed by it

- The **determinant** $|\,.\,|$ of a matrix yields the volume of a parallelepiped spanned by these vectors[3]



- $|J_T|$, *the Jacobian of* T, gives the change in volume at $\vec{a}$ due to T

Computing the PDF of a Transformation

- Let's try polar coordinates again: $\begin{pmatrix} x \\ y \end{pmatrix} = T\begin{pmatrix} r \\ \theta \end{pmatrix} = \begin{pmatrix} r\sin\theta \\ r\cos\theta \end{pmatrix}$

- $\left|\frac{\partial T\binom{r}{\theta}}{\partial\binom{r}{\theta}}\right| = |J_T| = \left|\begin{pmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} \end{pmatrix}\right| = \left|\begin{pmatrix} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{pmatrix}\right| = \mathrm{r}$

- $p(x,y) = \frac{p(r,\theta)}{r}$, or $p(r,\theta) = r\,p(x,y)$, which tells us: the change in probability density from (r,θ) to (x,y) is **inverse proportional to $\boldsymbol{r}$**

Sampling Joint PDFs Correctly

- For independent variables, the joint PDF $p(x, y, \ldots)$ is $p_X(x)p_Y(y)\ldots$

- In general, this is an assumption that we should not rely on

- Furthermore, after a transformation, only the joint PDF is known

- The proper way to sample multiple variables X, Y is to compute
  - the *marginal density function* $p_X(x)$ of one
  - the *conditional density function* $p_Y(y|x)$ of the other

Marginal and Conditional Density Function

- Assume we have obtained the joint PDF $p(x, y)$ of variables X, Y with ranges $[a_X, b_X)$ and $[a_Y, b_Y)$
- In a 2D domain with X, Y we can think of $p_X(x)$ as the average density of $p(x, y)$ at a given x over all possible values y
- We can obtain the *marginal density function* for one of them by *integrating out* all the others, e.g.: $p_X(x) = \int_{a_Y}^{b_Y} p(x, y)\, dy$
- We can then find $p(y|x) = \frac{p(x,y)}{p_X(x)}$

Adding More Variables

- What to do for multiple variables, e.g. X, Y and Z?
  - Find first marginal density $p_X(x) = \int_{aZ}^{bZ} \int_{aY}^{bY} p(x, y, z)\, dy\, dz$
  - Find first conditional density $p_X(y, z|x) = \frac{p(x,y,z)}{p_X(x)}$
  - Find second marginal density $p_Y(y|x) = \int_{aZ}^{bZ} p(x, y, z)\, dz$
  - Find second conditional density $p_X(z|x, y) = \frac{p(y,z|x)}{p_Y(y|x)}$
  - Integrate + invert first marginal, first and second conditional densities
  - Sample each of them
  - Extend ad libitum to even more variables

Sampling the Unit Disk: The Formal Solution

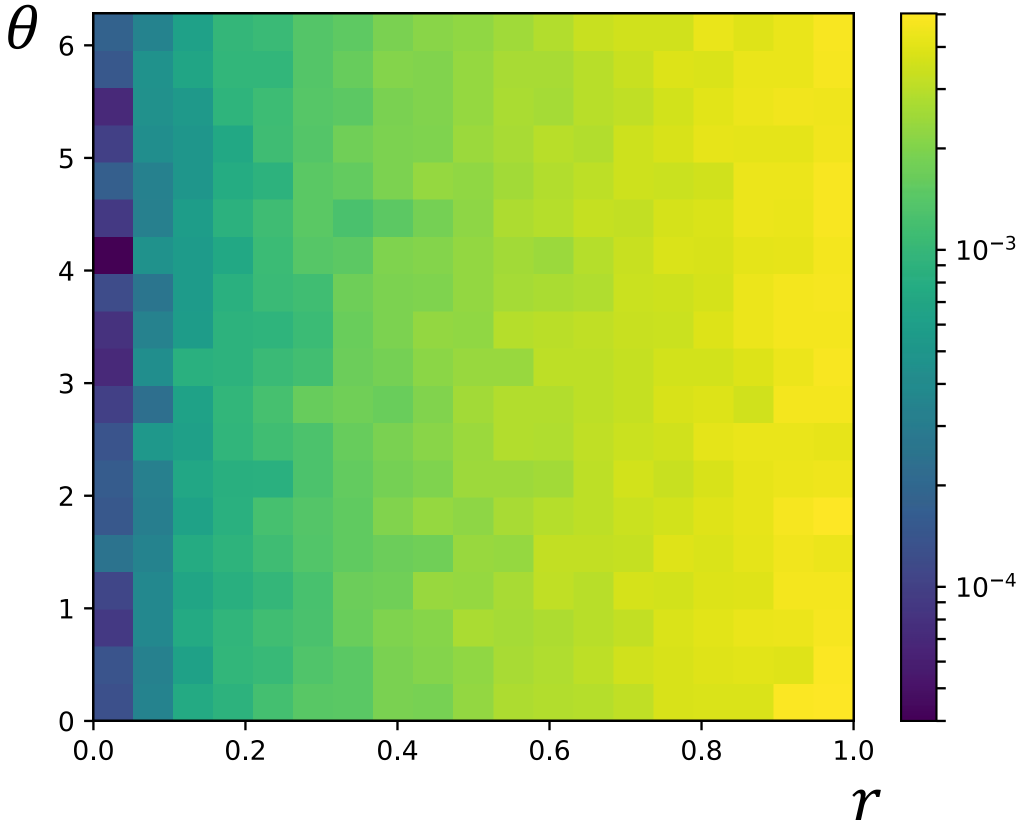
- We know the proportionality constant is π (area of sampled disk)

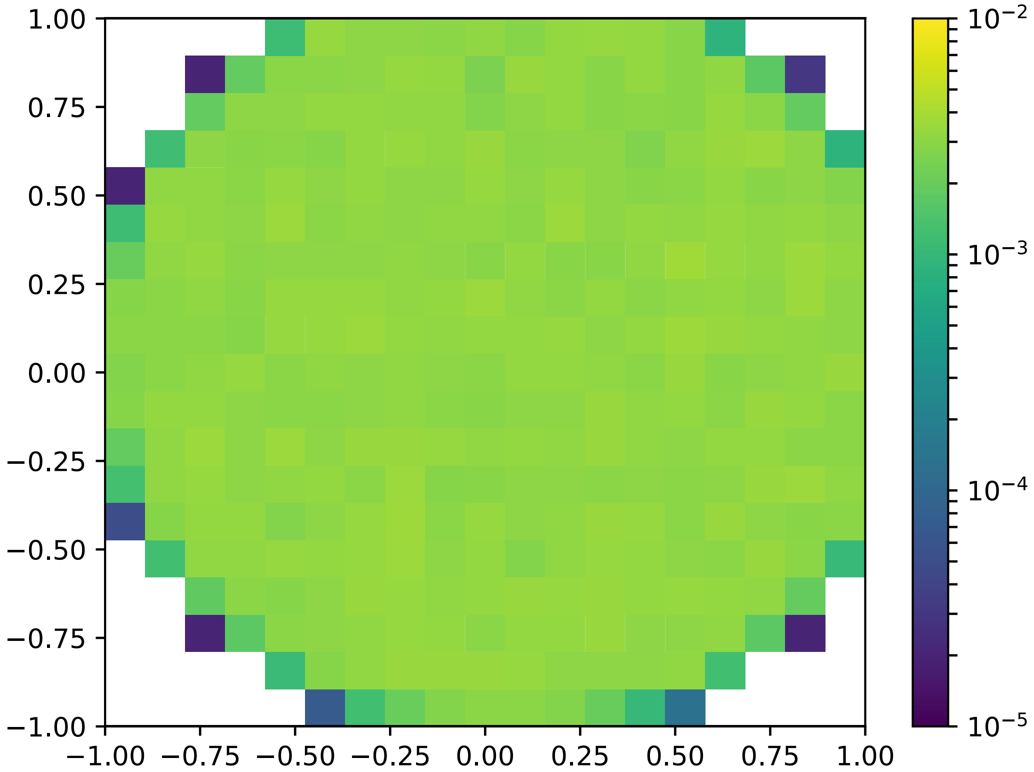
- Since we want uniform sampling and sample probabilities should integrate to 1, the target PDF in cartesian coordinates is $p(x, y) = \frac{1}{\pi}$

- $|J_T|$ told us that $p(r, \theta) = r\, p(x, y)$, so we want $p(r, \theta) = \frac{r}{\pi}$

- $p_R(r) = \int_0^{2\pi} p(r, \theta)\, d\theta = 2r$ and $p(\theta | r) = \frac{p(r,\theta)}{p_R(r)} = \frac{1}{2\pi}$

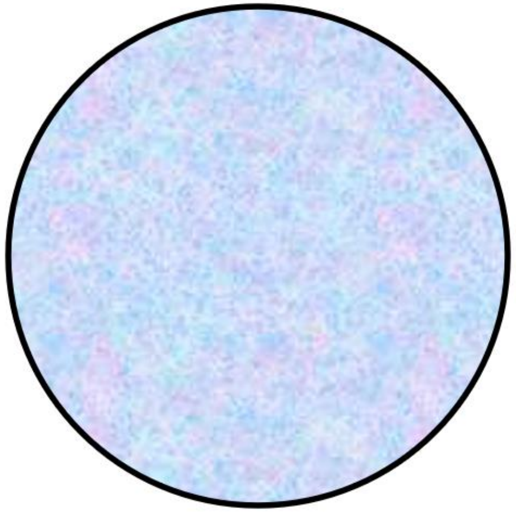
- If we create samples in polar coordinates for these PDFs, we will get the uniform distribution in (x, y) after applying transformation T





Sampling the Unit Disk: The Formal Solution

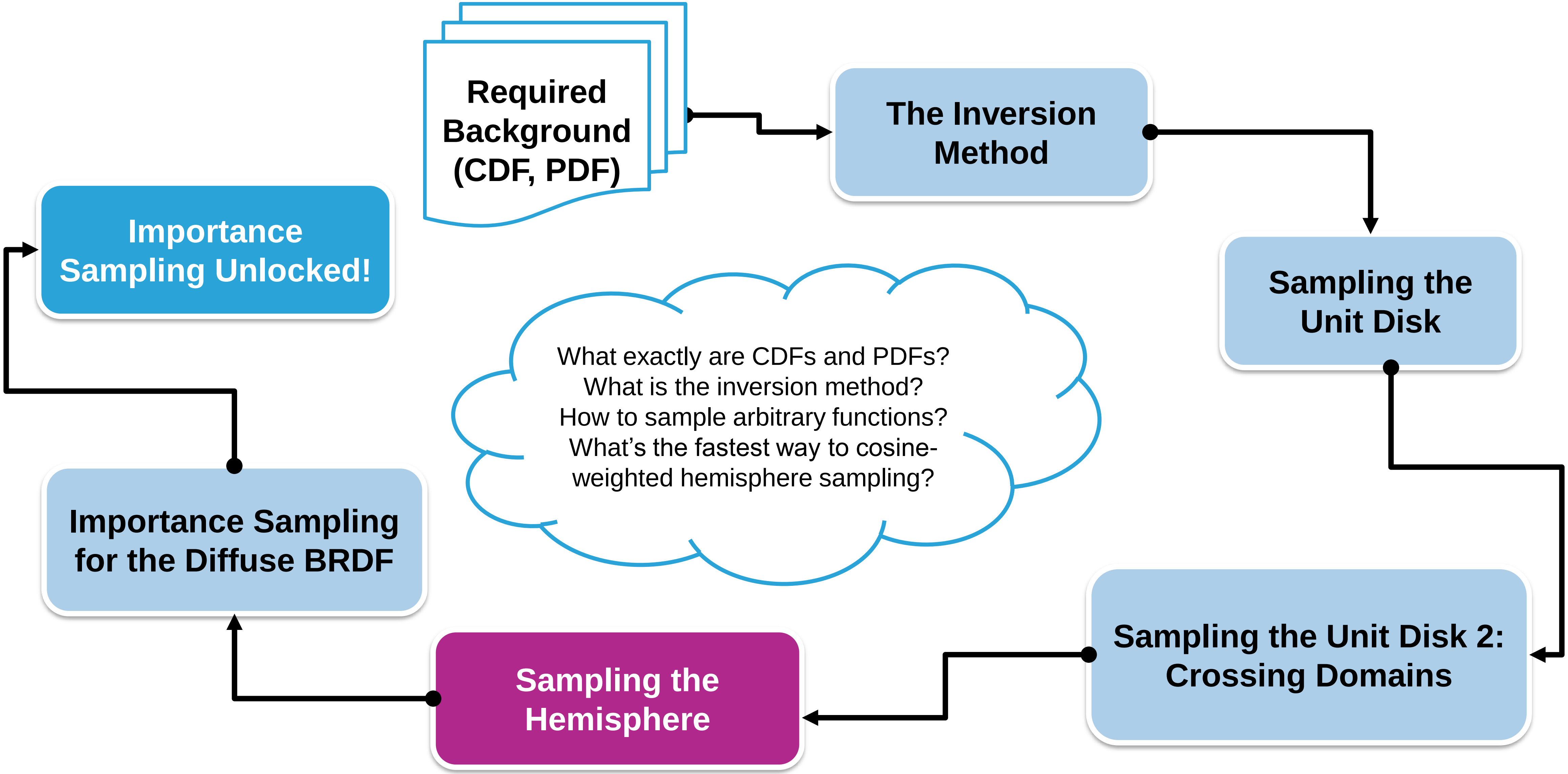
- Integrate marginal and conditional PDFs and invert—we get the same solution as before:
  - $r = \mathrm{P}_R^{-1}(\xi_1) = \sqrt{\xi_1}$
  - $\theta = P_\Theta^{-1}(\xi_2) = 2\pi\xi_2$



- $p(\theta|r)$ is constant: no matter what radius we are looking at, all positions on a ring of that radius (angle) should be equally likely

- Final integral: $RGB_{total} = \frac{\pi}{N}\sum_{i=1}^{N} RGB(R_i\, sin\, \Theta_i, R_i\, cos\, \Theta_i)$

Today's Roadmap



Moving on to the Hemisphere

- This took as a while, but we have seen all the formal procedures

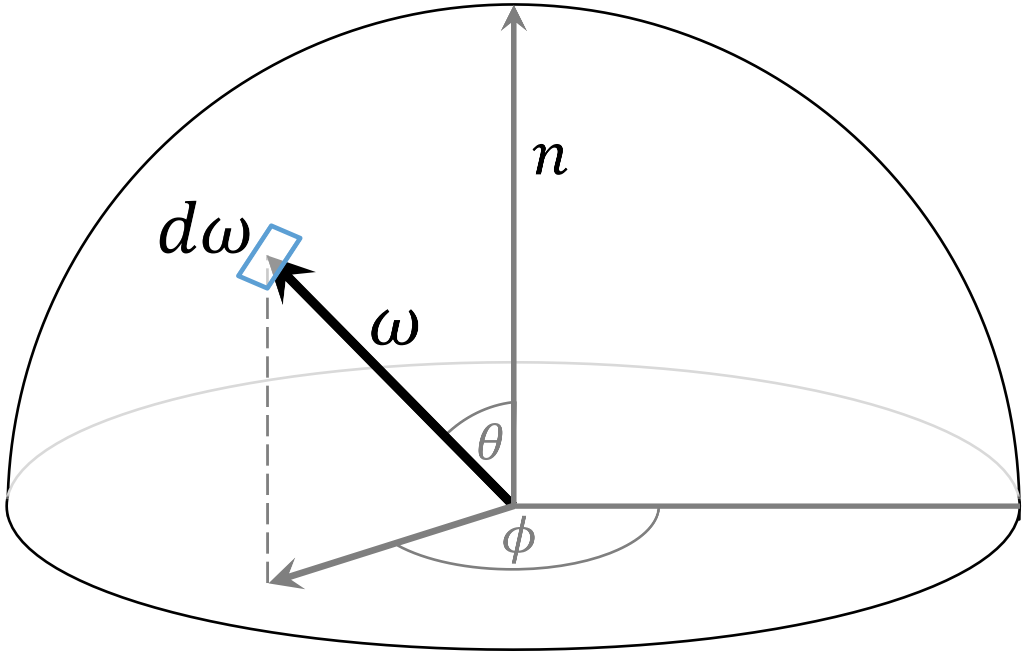
- We only need to switch from integrating planar area to points ω on hemisphere surface (i.e., vectors (x, y, z) with length 1)

- Use spherical coordinates and bijective T from (r, θ, ϕ) to (x, y, z):

$$x = r \sin\theta \cos\phi$$
$$y = r \sin\theta \sin\phi$$
$$z = r \cos\theta$$

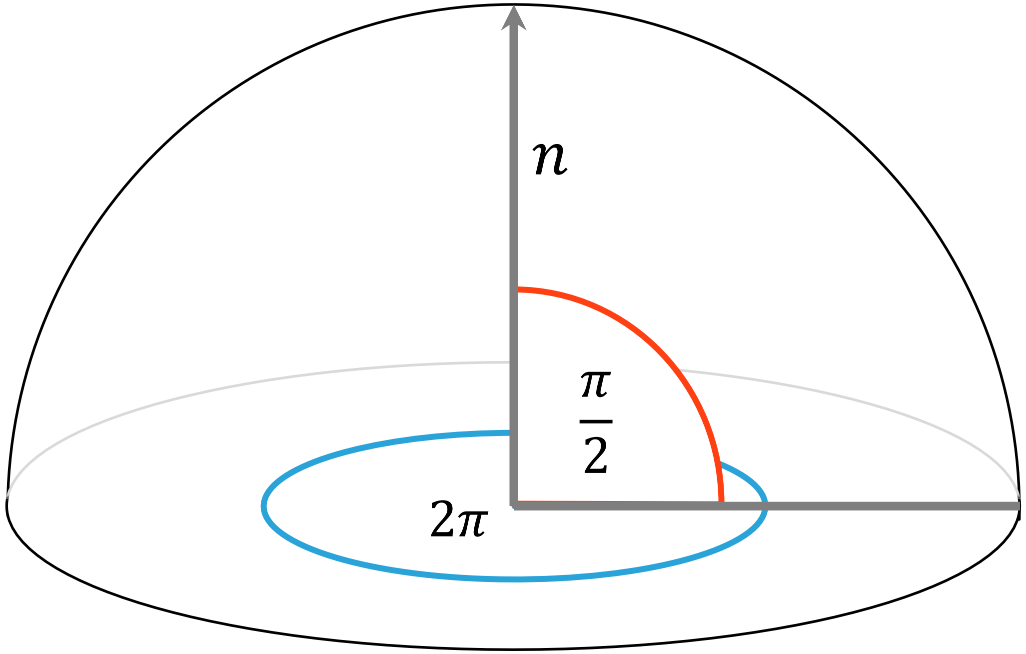
Deriving Integration Over Hemisphere

- Each direction ω represents an infinitesimal surface area portion $d\omega$
- How do we integrate a function $f(\omega)$ with differential $d\omega$?
- Integration over points on hemisphere surface ω, w.r.t. (θ, ϕ)



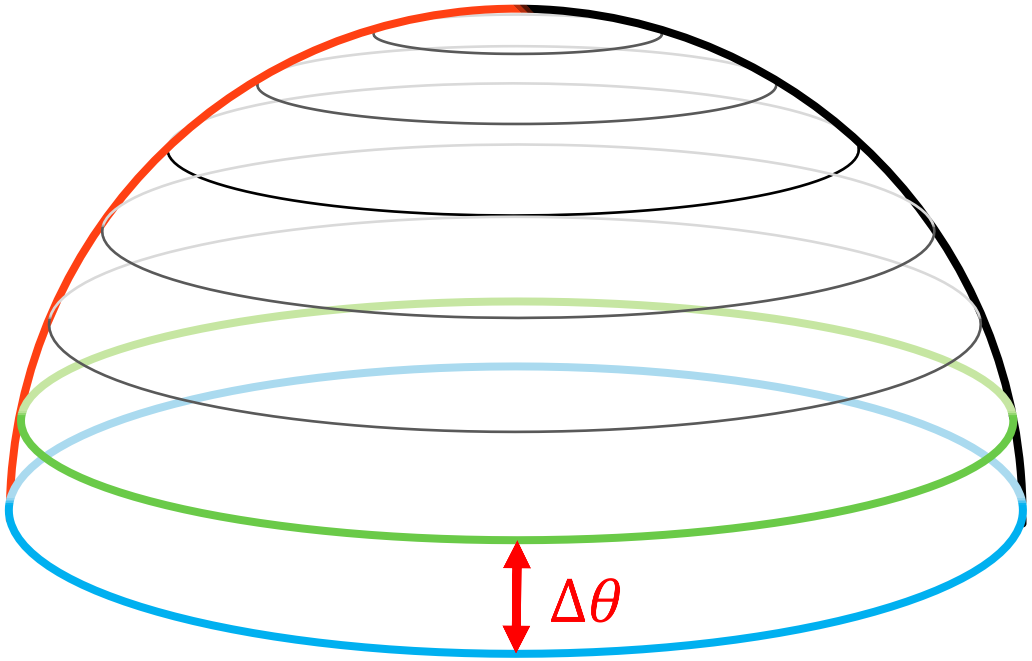
Deriving Integration Over Hemisphere

- We assume a planar surface with an upright facing normal n
- We use the integral intervals $\theta \in \left[0, \frac{\pi}{2}\right), \phi \in [0, 2\pi)$
- I.e., a curve from perpendicular to parallel for θ, a ring for ϕ



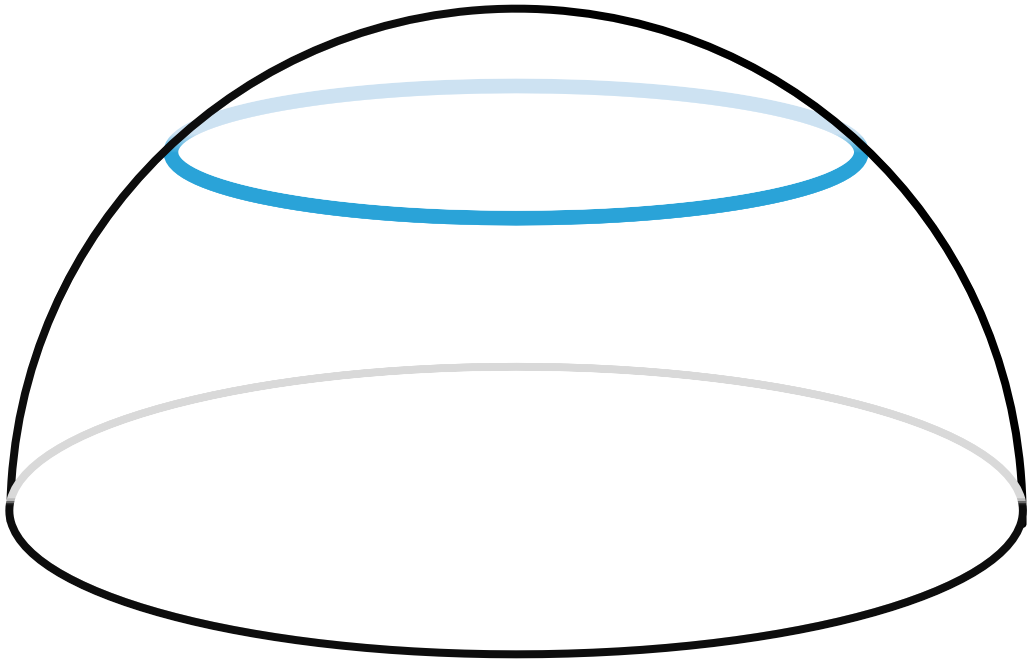
Deriving Integration Over Hemisphere

- We can split the surface along θ into ribbons of width $\Delta\theta \rightarrow d\theta$
- The upper edge of the ribbon is slightly shorter than the lower
- If we keep adding more and more ribbons, this difference vanishes



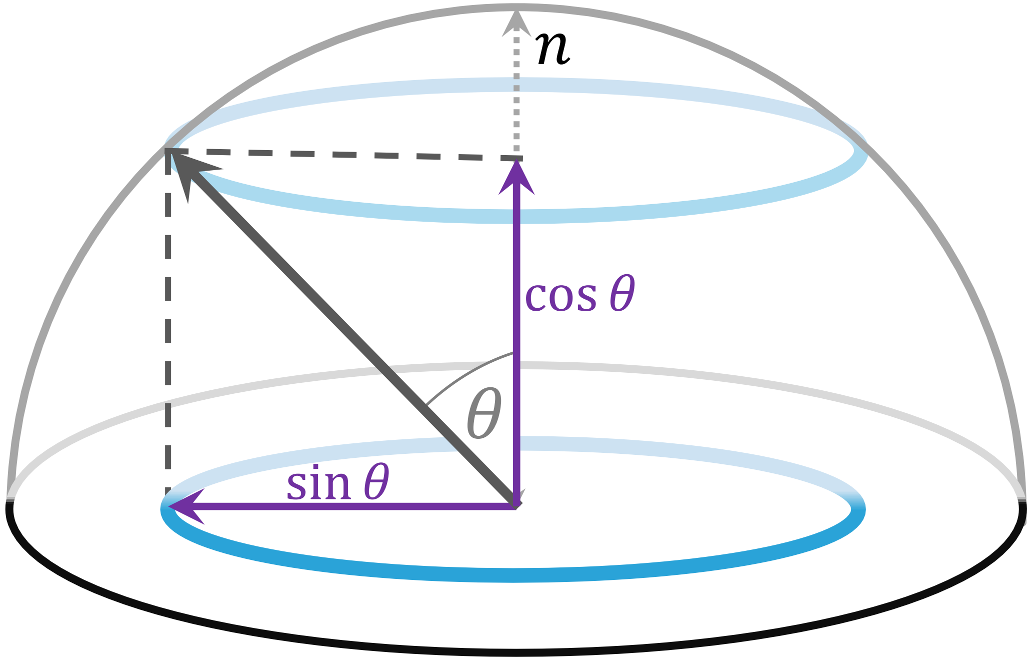
Deriving Integration Over Hemisphere

- As a ribbon's width goes to $d\theta$, its area becomes its length times $d\theta$
- We can find this length by projecting the ribbon to the ground
- Using trigonometry, we find the length of a ribbon is $2\pi \sin\theta$



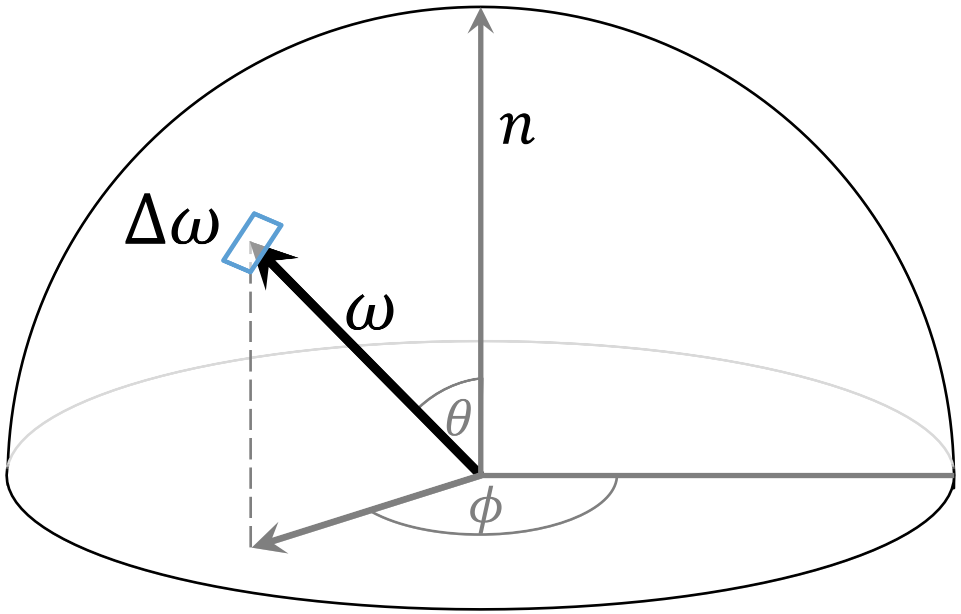
Deriving Integration Over Hemisphere

- As a ribbon's width goes to $d\theta$, its area becomes its length times $d\theta$
- We can find this length by projecting the ribbon to the ground
- Using basic trigonometry, we find the length of a ribbon is $2\pi \sin\theta$

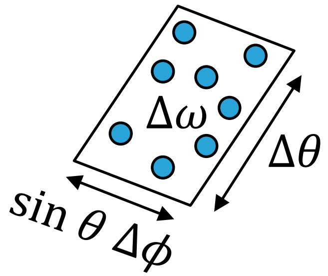


Deriving Integration Over Hemisphere

- The length of a ribbon spans the entire interval $\phi \in [0, 2\pi)$
- Convert the length to an integral over $d\phi$: $2\pi \sin\theta = \int_0^{2\pi} \sin\theta \, d\phi$
- The final integral: $\int_\Omega f(\omega)\, d\omega = \int_0^{\frac{\pi}{2}} \int_0^{2\pi} f(\omega) \sin\theta \, d\phi \, d\theta$



Deriving PDF for Hemisphere Sampling



- Integral of $f(\omega)$ over area $\Delta\omega = \int_{\Delta\omega} f(\omega)\,d\omega$
- Integral of $f(\omega)$ w.r.t. $(d\theta, d\phi) = \int_{\Delta\theta}\int_{\Delta\phi} f(\omega)\sin\theta\,d\phi\,d\theta$
- Integration domain and $f(\omega)$ are identical, thus: $d\omega = \sin\theta\,d\phi\,d\theta$
- $\omega \leftrightarrow (\theta, \phi)$ is bijective, we have $p(\theta, \phi)\,\mathrm{d}\theta\,d\phi = p(\omega)\,d\omega$ and:

$$p(\theta, \phi) = \sin\theta\,p(\omega)$$

Deriving PDF for Hemisphere Sampling, the Formal Way

- Target distribution in ω, which is (x, y, z) with $\sqrt{\mathrm{x}^2 + y^2 + z^2} = 1$

- The transformation T from (r, θ, ϕ) to (x, y, z):

$$x = r \sin\theta \cos\phi$$
$$y = r \sin\theta \sin\phi$$
$$z = r \cos\theta$$

- The Jacobian of the transformation T gives $|J_T| = r^2 \sin\theta$

- $r = 1$, so we have $p(1, \theta, \phi) = \sin\theta \, p(x, y, z) = \sin\theta \, p(\omega)$

Uniformly Sampling the Unit Hemisphere

- The domain, i.e., the unit hemisphere surface area, is 2π.
  Uniformly sampling the domain over ω implies $p(\omega) = \frac{1}{2\pi}$

- Hence, since $p(1, \theta, \phi) = \sin\theta\ p(\omega)$, we want $p(\theta, \phi) = \frac{\sin\theta}{2\pi}$

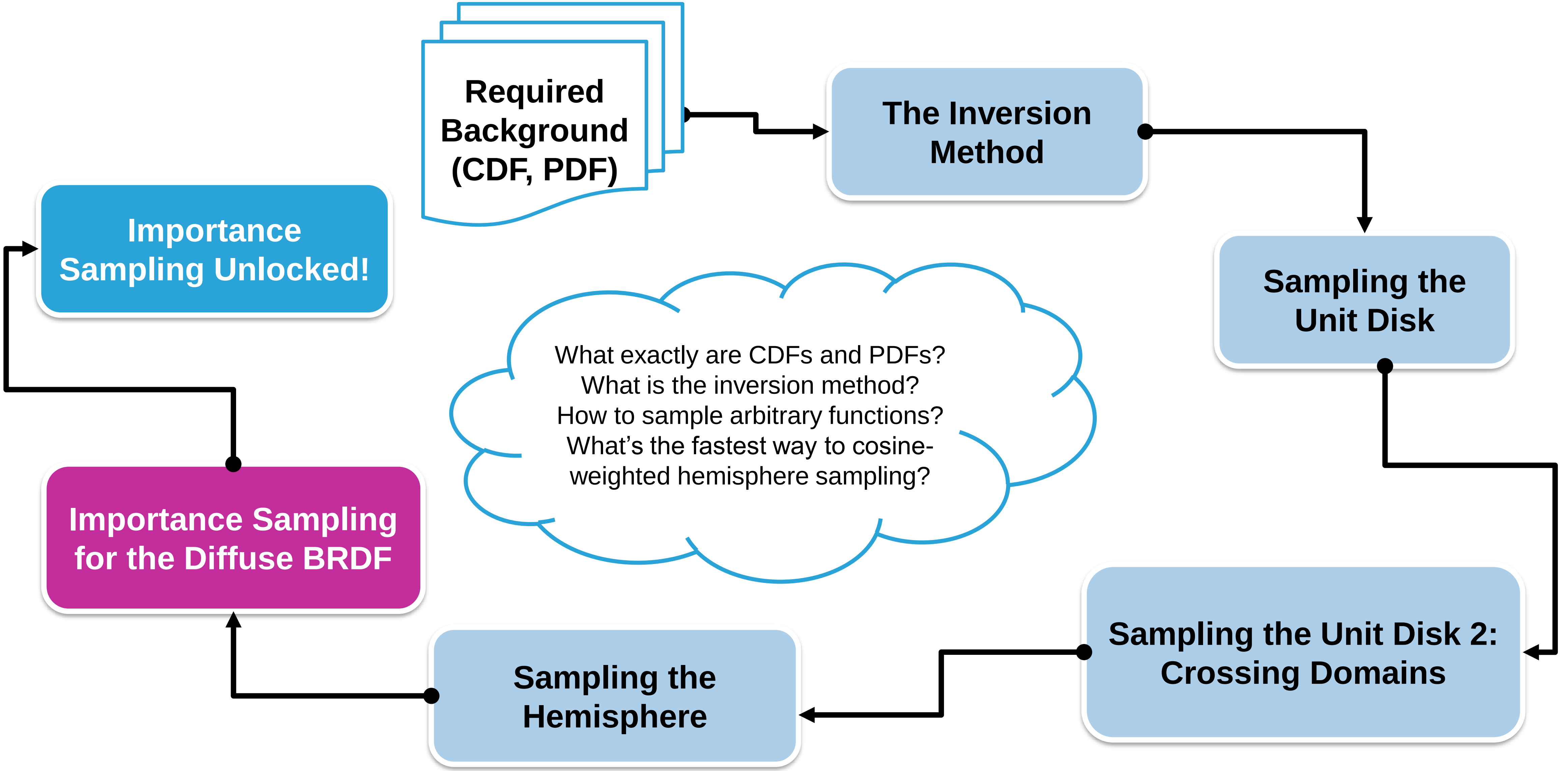
- Marginal density $p_\Theta(\theta)$: $\int_0^{2\pi} p(\theta, \phi)\, d\phi = \sin\theta$

- Conditional density $p(\phi|\theta)$: $\frac{p(\theta,\phi)}{p_\Theta(\theta)} = \frac{1}{2\pi}$

Uniformly Sampling the Unit Hemisphere – Complete

- Antiderivative of $p_\Theta(\theta)$: $\int \sin\theta \, d\theta = 1 - \cos\theta$ (added constant 1)
- Antiderivative of $p(\phi|\theta)$: $\int \frac{1}{2\pi} \, d\phi = \frac{\phi}{2\pi}$
- Invert them to get $\theta = \cos^{-1}\xi_1$ (cos is symmetric), $\phi = 2\pi\xi_2$
- Apply transformation T on (θ, ϕ) to obtain uniformly distributed ω
- Finally done!

Today’s Roadmap

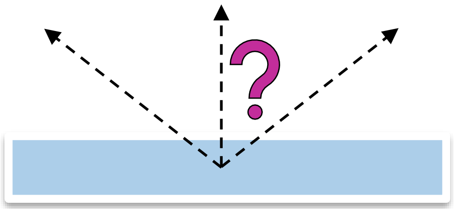


Importance Sampling the Diffuse BRDF

- Let‘s look once more at the reflected light in the rendering equation

$$\int_{\Omega} \underbrace{f_r(x, \omega \rightarrow v) L(x \leftarrow \omega) \cos(\theta_\omega)}_{f(x)} \, d\omega$$

- When we bounce at a point x, we already know quite a bit:
  - If we use a diffuse BRDF, then $f(x, \omega \rightarrow v)$ is a constant factor $\frac{\rho}{\pi}$
  - We can predict the cosine term—it depends on our choice of ω
  - The tricky part, the big unknown, is the $L(x \leftarrow \omega)$
  - Which directions will indirect light come from?



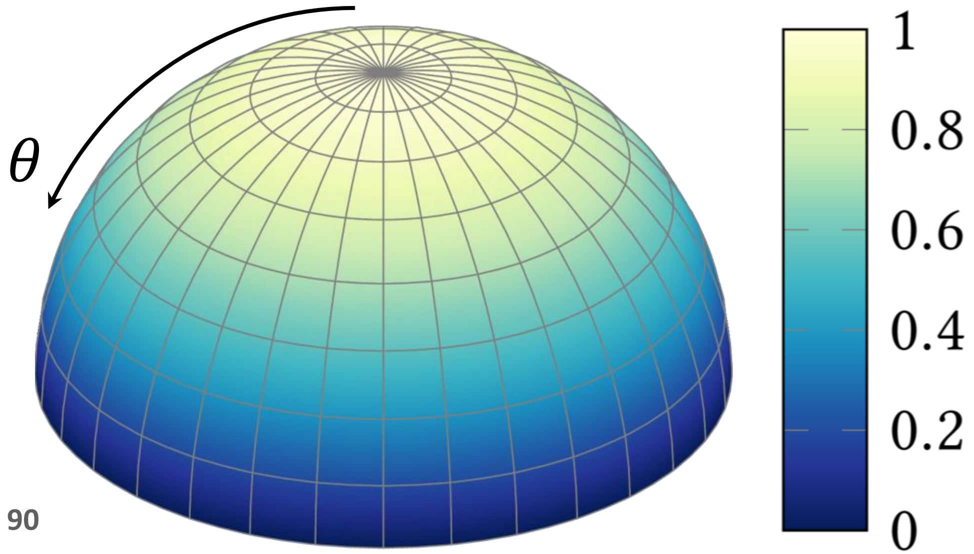
Importance Sampling the Diffuse BRDF

- If we don‘t know anything about L, let‘s just assume a constant k

$$\int_{\Omega} \frac{\rho}{\pi} k \cos(\theta_\omega)\, d\omega$$

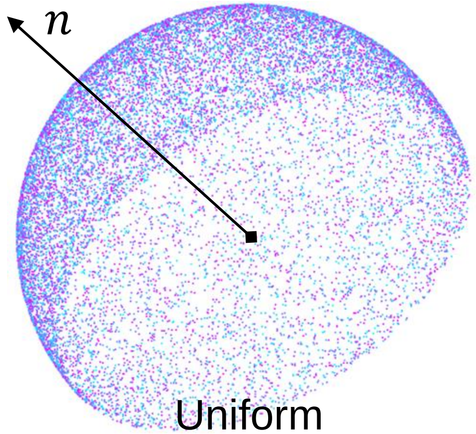
- $\frac{\rho}{\pi}$ and k are constant, so clearly, $\frac{\rho}{\pi} k \cos(\theta_\omega) \propto \cos(\theta_\omega)$

- With these assumptions, the integrand function is governed entirely by the term $\boldsymbol{cos(\theta)}$!

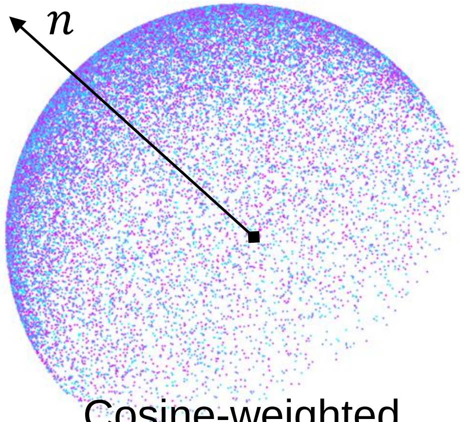


Importance Sampling the Diffuse BRDF

- We know that the ideal distribution $p(x)$ for importance sampling a function $f(x)$ is the one that minimizes variance, i.e., $\propto f(x)$ itself
- With the assumption of constant light from all directions, our integrand $f(x)$ was simplified to something proportional to $\cos(\theta)$
- Idea: Importance-sample hemispheres around hit points with diffuse materials with distribution $p(\omega) \propto \cos(\theta_\omega)$



Uniform hemisphere sampling



Cosine-weighted hemisphere sampling

Cosine-Weighted Hemisphere Sampling?

- In the first half, we saw how you can apply the inversion method for sampling arbitrary distributions

- In the second half, we were all about making sure that we can reach our target distribution when we move from one domain to another

- Cosine-weighted hemisphere sampling is a combination of the two

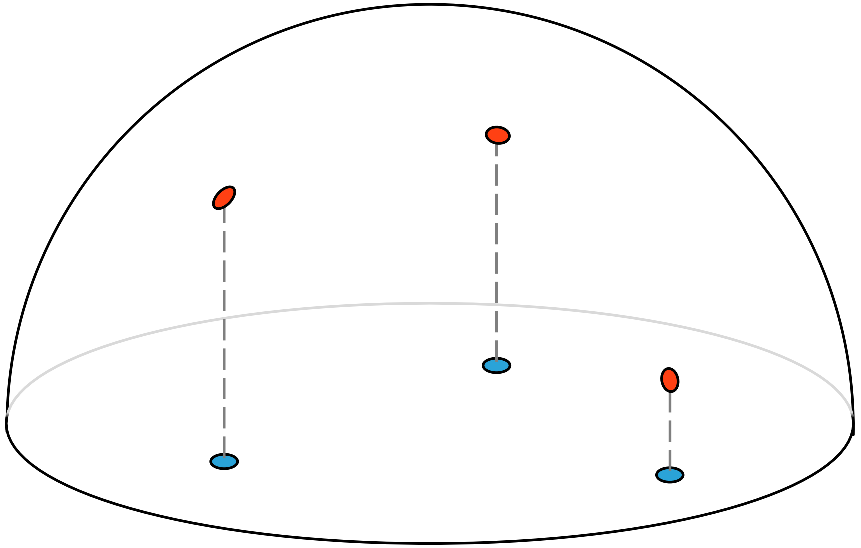
- We have gone through all the necessary steps.
  <u>Try to solve this formally with the inversion method as an exercise</u>!

Cosine-Weighted Hemisphere Sampling!

- Malley's method: uniformly pick (x, y) samples on the unit disk

- Project them to the hemisphere surface $\left(z = \sqrt{1 - x^2 - y^2}\right)$

- $p(\omega) = \dfrac{\cos\theta}{\pi}$



- Done! Your samples are now distributed with $p(\omega) \propto \cos\theta$! (Why? And why does this work? Try to find your own proof!)

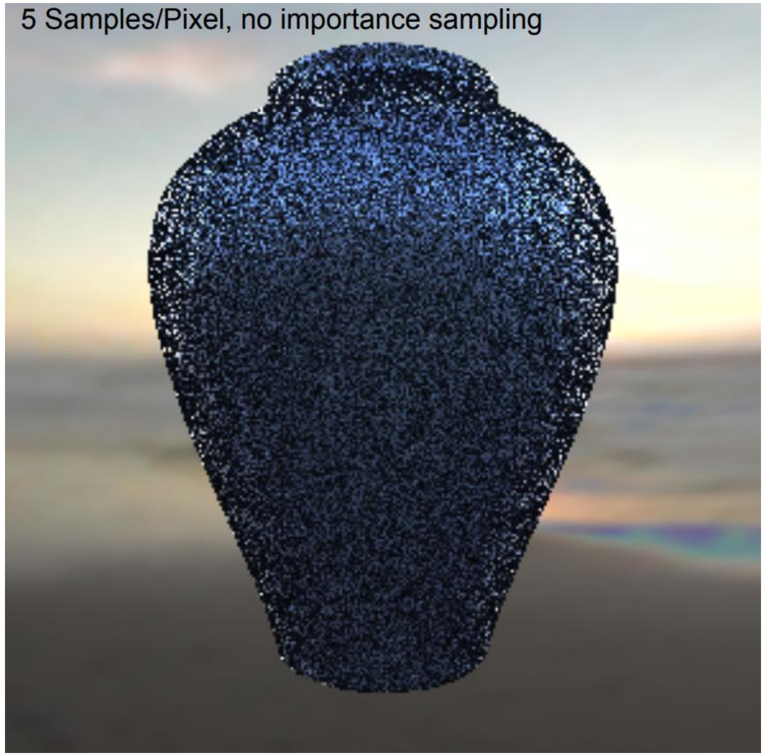
Importance Sampling the Diffuse BRDF

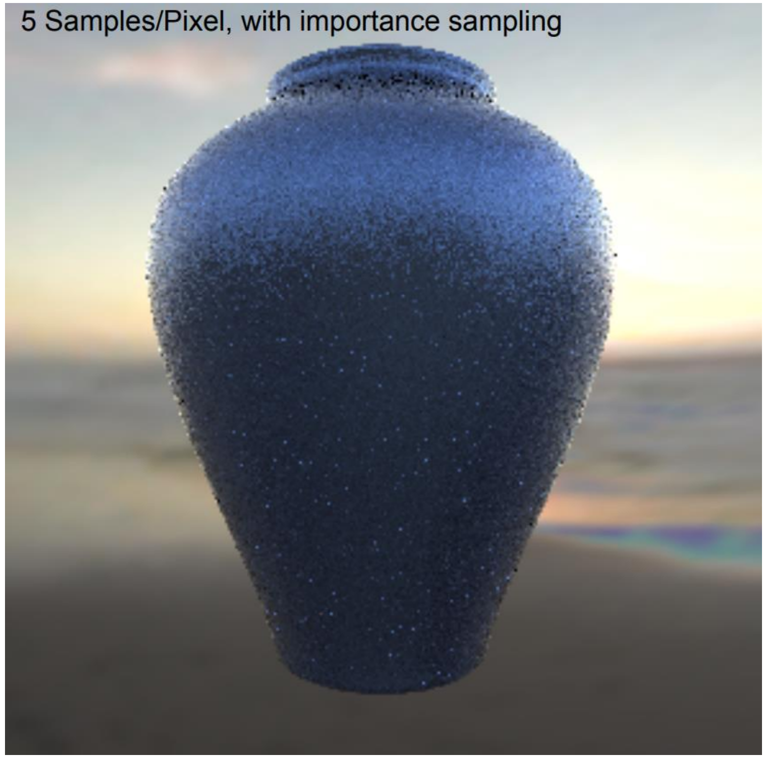
AO, 64 samples, uniform hemisphere sampling

AO, 64 samples, diffuse BRDF importance sampling

More Importance Sampling

- The impact will be much greater when we add non-diffuse materials
- BRDF functions can be rather complex…
- …but can often be nicely approximated

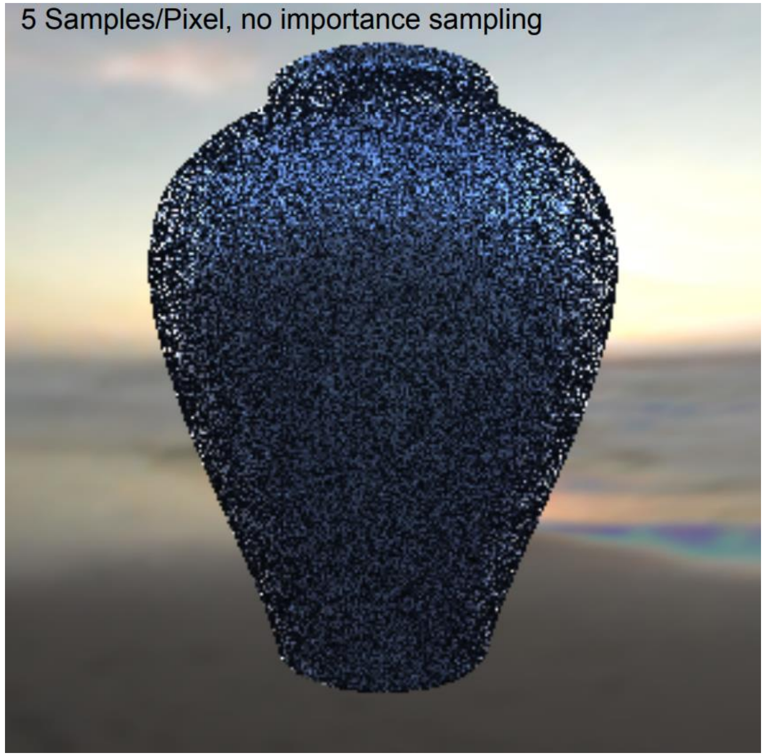


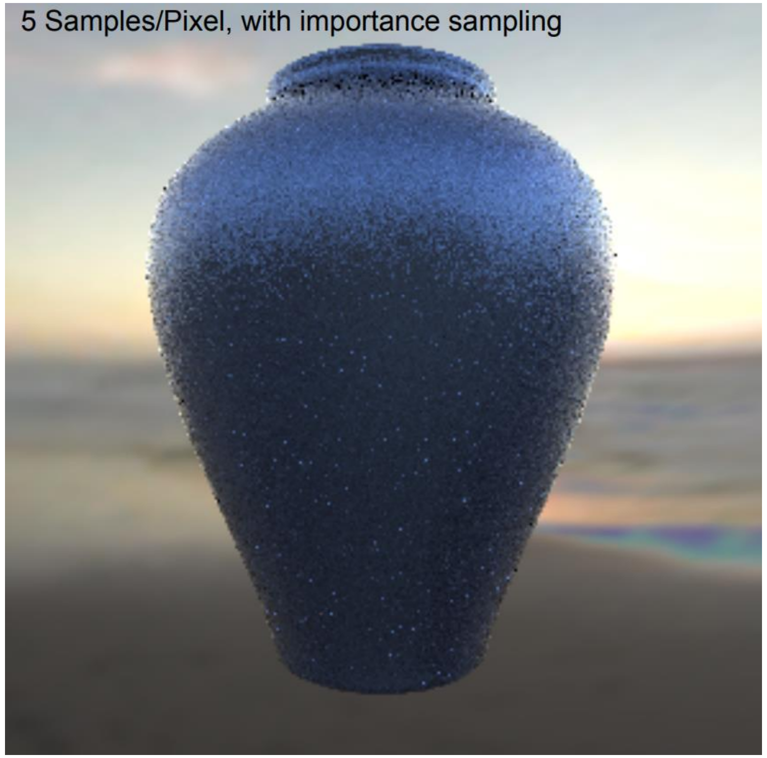


- You will want to sample with distributions more complex than $\cos\theta$

More Importance Sampling

- Consider the modified Beckmann distribution for microfacet BRDFs
- $D(\theta,\phi) = \dfrac{e^{\frac{-\tan^2\theta}{\alpha^2}}}{\pi\alpha^2\cos^3\theta}$
- Yes, seriously!





- Good luck with intuitive reasoning! Challenging, but doable task with basic trigonometric identities and the inversion method!

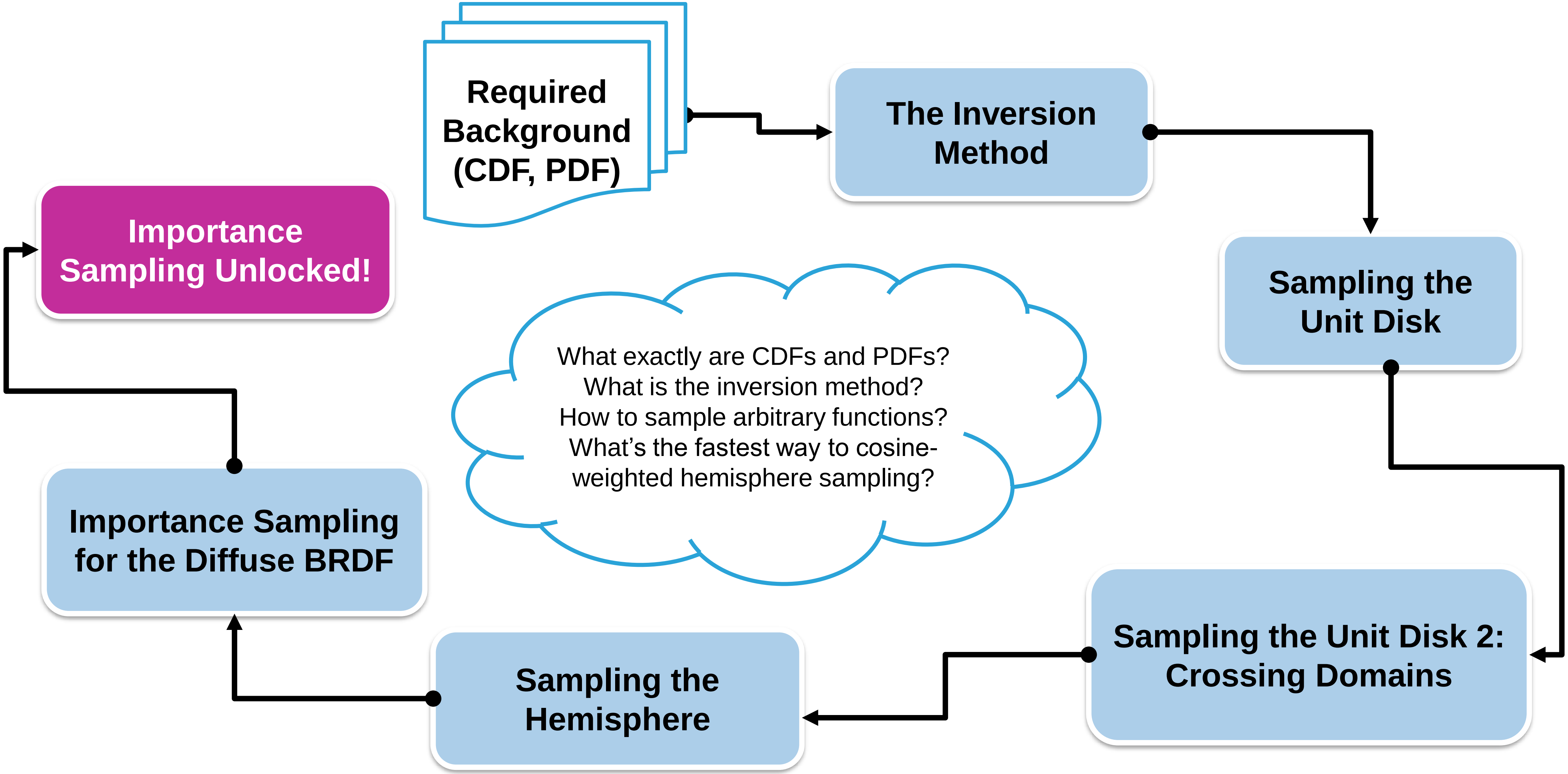
Importance Sampling the Full Rendering Equation

- As you can imagine, this is a much more complex task
- In fact, an enormous amount of research in rendering is actively pursuing better and better ways to make this happen
- Other sophisticated methods, like multiple importance sampling (MIS), can be of great help here!
- We will hear more about MIS in upcoming lectures…

Importance Sampling Summary

- If we do Monte Carlo integration of $f(x)$, it's best to use a sample distribution $p(x)$ that closely mimics $f(x)$
- For a desired $p(x) \propto f(x)$, we can use the **inversion method** to get the methods for generating samples and probability densities
- If you cannot turn $f(x)$ into a valid PDF, try to find a close match
- When we transform samples between domains, we have to make sure they have the desired distribution in the target domain!

Today’s Roadmap



References and Further Reading

- Slide set based mostly on chapter 13 of *Physically Based Rendering: From Theory to Implementation*
- [1] Steven Strogatz, *Infinite Powers: How Calculus Reveals the Secrets of the Universe*
- [2] Video, *Why "probability of 0" does not mean "impossible" | Probabilities of probabilities, part 2*: https://www.youtube.com/watch?v=ZA4JkHKZM50
- [3] Video, *The determinant | Essence of linear algebra, chapter 6*: https://www.youtube.com/watch?v=Ip3X9LOh2dk
- [4] SIGGRAPH 2012 Course: *Advanced (Quasi-) Monte Carlo Methods for Image Synthesis*, https://sites.google.com/site/qmcrendering/
- [5] Wikipedia, *Volume Element*, https://en.wikipedia.org/wiki/Volume_element