Introduction to Random Walks

[see -> BSSRDF->TheoreticalFoundation->dEon->Zero-Variance Theory for Efficient Subsurface Scattering]

We'll see in a bit that with absorption, scattering and phase functions we can model the appearance of translucent materials. The unbiased approach to this, is a brute-force random walk. Basically the whole process is to :

- sample a free-path-length distribution (i.e. to move the walk one step)
- sample a phase function (i.e. changing walk direction)
- until we escape at boundary where eventually we sample its BRDF

Note here we don't take into account boundary conditions like Fresnel, that simply means we pretend there's no interaction between our random walks and the boundary itself if not for sampling the boundary BSDF.. proper interactions would account for example for the walk that could get back into the medium once reached the boundary.

[Note that however Fresnel may still play a role with the BSSRDF layer as generally the diffuse contribution, to obey energy conservation, is multiplied by 1-Fresnel]

So what are we doing here ? Considering light as a flux of particles (photon beam) that interact with other particles in a medium. This is the **photonic interpretation of electromagnetic scattering** by macroscopic particles and has its roots in Einstein's paper on the photoelectric effect (1905).. where photons are considered energy quanta localized at points in space and that behaves as complete units.. and are these phenomenological photons that form the incident beam of light that change direction upon scattering on the macroscopic particles or just disappear due to absorption.

[see -> BSSRDF->TheoreticalFoundation->Gustav Mie and the Fundamental Concept of Electromagnetic Scattering by Particles]

Furthermore in Mie theory (1908), **the scattering object** is defined as a finite volume with a refractive index different from that of the surrounding infinite homogeneous medium. And so on with the traditional intuitive perception of scattering seen as a plane electromagnetic wave propagating in an infinite non-absorbing medium without a change in its intensity until it will be in the presence of particles, so that the difference between the total field in the presence of the particle and the original field that would exist in the absence of the particle might be thought of as the field scattered by the particle.. in other words, the total field in the presence of the particle is represented as the sum of the respective incident (original) and scattered fields.. thus, it is the modification of the total electromagnetic field caused by the presence of the particle that is **called electromagnetic scattering**.

Now here, what do we mean with, - phenomenological ? That can be said to be **a physical phenomenon**. So can we say that electromagnetic scattering is a physical phenomenon ? Of course, - we

would say. But. It is not however to be treated as a solitary physical process. So the question that belongs to this could be, - where does it end the physics and starts the mathematics ?

[see -> BSSRDF->TheoreticalFoundation->dEon->An Hitchhiker's Guide to Multiple Scattering]

Once physics has been used to derive the distributions that govern the random flight of one particle, the physics ends and **the transport theory begins**. Once we have completely defined the properties of a scattering problem, we begin the core statistical analysis of considering a point particle with initial position and direction drawn randomly from the source distribution. The particle then undergoes an alternating sequence of straight displacements and scattering events. **The distances between scattering events and the deviations in angle at each scattering event are random variables drawn from given distributions.** The process is eventually terminated when the particle is randomly absorbed, or when the particle escapes the finite system.

So let's highlight some strong physical points about scattering and our particles.
We limit our attention here to **linear scattering** :
- particles do not interact with each other and with other particles in the walk
- particles do not alter the nature of the medium they are scattering/absorbing into

A linear transport problem is defined by :
- specifying a medium/scene,
- its properties and boundaries,
- a set of light sources.
- and a detector sensitivity over the phase space (rendering camera)

What follows belongs to **classical 'analog' sampling** of random walks, for that we employ this estimator :
- a particle arrives at the boundary entering the medium along a cosine direction μi
- initial particle weight is 1
- sample direction with phase fnc
- sample displacement with exponential law
- check if the particle last step did cross the boundary
- if not, calculate its new weight and sample a new dir and step
- if yes sample boundary BSDF

Material Properties

Absorption, Scattering and Phase functions

[see -> Reflection from Layered Surfaces due to Subsurface Scattering]

The surface of skin, leaves, snow, paint etc. is comprised of one or more layers composed of a mixture of randomly distributed particles or inhomogeneities embedded in a translucent media. Particle distributions can also exist, in which case the material properties are given by the product of each particle's properties times the number of particles per unit volume.

The intensity of the backscattered and transmitted light depends on the absorption and scattering properties of the material. These **cross sections** are interpreted as the probability per unit length of an interaction of a particular type. In classical physics, this probability often converges to a deterministic proportion of excitation energy involved in the process, so that, the cross section specifies the amount of optical power scattered from light of a given irradiance.

σ_a= absorption coefficent
σ_s= scattering coefficent
$\sigma_t = \sigma_a + \sigma_s$ = extinction coefficent

These are physical units so they should be used with a scaling parameter, i.e. for meters use a x1000.

The **mean free path** is equal to the reciprocal of the total cross section, $1/\sigma_t$.
An important quantity is the **albedo**, which equals W = σs/σt. If the albedo is close to 1, the scattering cross section is much greater than the absorption cross section, whereas if the albedo is close to 0, absorption is much more likely than scattering. That's also why the albedo is considered as the scattering efficency factor.

However while there're measurement of 'sigmas' (aka, optical parameters) that will approach specific materials like grapefruit potato skin leaves etc. [see -> A Practical Model for Subsurface Light Transport, fig.5; for Skin, see -> Tissue Optics: Light Scattering Methods and Instruments for Medical Diagnosis; for Leaves, see -> Fukshansky et al., 1993; both mentioned in -> Light Diffusion in Multi-Layered Translucent Materials] it is really cumbersome to rely on those when one wants some artistic control. So we generally remap Albedo and MeanFreePath.. ie. SSS color and SSS radius to the above quatities with the following :

In[95]:=
```
(* see ->
 Practical and Controllable Subsurface Scattering for Production Path Tracing *)
remapVolToScatter[albedo_, radius_] := Module[{a = albedo, d = radius},
   α = 1 - e^(-5.09406*a+2.61188*a^2-4.31805*a^3);
   s = 1.9 - a + 3.5 (a - 0.8)^2;
   σ = 1.0 / (d * s);
   x = σ * α;
   {σ, x, α} (*--gives back sigmaT, sigmaS and single-scattering Alpha --*)
  ];
```

In[96]:=
```
rgbAlbedo = List[0.5, 0.75, 0.3];
rgbRadius = List[1, 0.3, 0.2];
remapVolToScatter[rgbAlbedo, rgbRadius]
```

Out[98]=
```
{{0.58309, 2.87666, 2.0202},
 {0.531952, 2.83235, 1.52685}, {0.912298, 0.984595, 0.755792}}
```

Now, let's analyse what we get back as optical parameters when our albedo is mid grey.. ie. the characterization of sigmas will be mainly from the mfp.

In[99]:=
```
rgbAlbedo = List[0.5, 0.5, 0.5];
rgbRadius = List[1, 0.3, 0.2];
remapVolToScatter[rgbAlbedo, rgbRadius]
```

Out[101]=
```
{{0.58309, 1.94363, 2.91545},
 {0.531952, 1.77317, 2.65976}, {0.912298, 0.912298, 0.912298}}
```

Let's look at the Red channel for example, it is 1. What we see is that we have higher Greens and Blues than Red which has 1 as mfp. What this means ? That the Greens and Blues will be absorbed/scattered more than Reds. Each component of these coefficients indicates how much Red, Blue and Green is absorbed and scattered by the medium.

So we have two main material properties here: absorption and scattering. These apply to all objects. When an object is black it absorbs almost all the light. If an object is white, it absorbs very little light and scatters most of it back. We may think that only absorption accounts for the darkening of the material but effectively also scattering does .. when a light is scattered not all light will reach our eyes because its directions are deflected to other paths.

Generally with 'scattering' we intend a surface material property, aka **surface scattering** which is generally modelled by a BRDF. Here instead we will deal with **subsurface scattering**. However under a general definition this is the same as surface scattering.. it is just that particles substrate at the surface will also absorb light so the 'surface' is going deeper into the material and as a result of that it won't scatter back necessarly from the same point it did hit the material originally. So we can consider a **BRDF** an approximation (where $x_o = x_i$) of a full **BSSRDF** and formalize the BSSRDF definition as the ratio between outgoing radiance at x_o, as a result of incoming differential radiant flux $d\Phi$ at x_i.

[see -> A Practical Model for Subsurface Light Transport and Skin Rendering: Reflectance and Integration]

Because we're interested in the outgoing radiance that will hit back our eyes or camera sensor, given a **BSSRDF**, the outgoing radiance L_o is computed by integrating the incident radiance L_i over incoming direction and area :

$$L_o\left(x_o, \vec{w_o}\right) = \int_A \int_{2\pi} S\left(x_i, \vec{w}_i ; x_o, \vec{w_o}\right) L_i\left(x_i, \vec{w}_i\right) \left(\vec{n}.\vec{w}_i\right) dw_i\, dA\left(x_i\right)$$

Note the x_i, x_o where for a BRDF $x_o = x_i$;

S(·) is the BSSRDF : $\frac{dL\left(x_o, \vec{w}_i\right)}{d\phi\left(x_i, \vec{w}_o\right)}$

Note from the above $L_o(\cdot)$, the cosine term $\vec{n}.\vec{w}_i$ diminishes the density of photons arriving within dA for large angles ..

$$\frac{d^2\,\Phi}{dA\, d\vec{w}} = \left(\vec{w}\cdot\vec{n}\right) L\left(x, \vec{w}\right)$$

aka, the flux received at xi within a differential area, dA, from a direction within a differential angle around $\vec{w}_i$

The BSSRDF S relates the differential outgoing radiance at x_o to the differential incident flux at x_i as :

$$dL_o\left(x_o, \vec{w_o}\right) = S\left(x_i, \vec{w}_i, x_o, \vec{w_o}\right) d\Phi_i\left(x_i, \vec{w}_i\right)$$

Light propagation in a particinpating medium is described by the radiative transport equation (RTE), aka volume rendering linear integro-differential equation (VRE):

$$\left(\vec{w}\cdot\vec{\nabla}\right) L\left(x, \vec{w}\right) = -\sigma_t L\left(x, \vec{w}\right) + \sigma_s \int_{4\pi} p\left(\vec{w}, \vec{w'}\right) dw'$$

[see -> Skin Rendering: Reflectance and Integration]

The equation describes the rate of change in radiance.

Where σ_t and σ_s are our extinction and scattering coefficents. The quantity of photons scattered per unit traveled is given by $\sigma_s(x)$ and the quantity absorbed is given by $\sigma_a(x)$. The rate of loss in radiance flowing in the direction $-\vec{\omega}$ at x is accounted for by the first term on the right side of equation. The second term accounts for gain in radiance due to in-scattering. This is done by integrating the radiance over all possible directions. During integration the radiance is scaled by $\sigma_s(x)$ and the phase function $p(x, \vec{w}', \vec{w})$. The former will give us the portion of radiance from $\vec{w}'$ scattered at x and the latter will give us the portion of this contribution which scatters specifically into the direction $\vec{w}$.

Where $\int_{4\pi} p\left(\vec{w}, \vec{w'}\right)$ is the **phase function** that obeys $\int_{4\pi} p\left(x, \vec{w}', \vec{w}\right) d\vec{w}' = 1$.

That is a function describing the dependence of scattered radiance on scattering angle. The phase

function is a dimensionless and normalized version of the scattering function, such that the integral of the phase function over 4π steradians is unity. The phase function for a given wavelength is a property of the medium, not of the incident radiation.

The phase function is the angular distribution of light intensity scattered by a particle at a given wavelength. It is given at an angle θ which is relative to the incident beam. The phase function is the intensity (radiance) at θ relative to the normalized integral of the scattered intensity at all angles. It is defined as :

$$P(\theta) = \frac{F(\theta)}{\int_0^{\pi} F(\theta) \sin\theta \, d\theta}$$

where F is the radiance and P(θ) is the probability of a photon being scattered in a particular direction θ. That also means that when integrated over the sphere it must equal zero, thing known as the '**normalization condition**' and shown by :

$$\frac{1}{4\pi} \int_0^{2\pi} \int_0^{\pi} p(\cos\theta) \sin\theta \, d\theta \, d\phi$$

Note that the azimuthal dependence ϕ of P is often removed, which is possible under the assumption of a spherical particle. Eventually as there're many BRDFs for surface scattering, there're many phase functions for subsurface scatter.

Phase fncs share a similarity with BRDFs in that phase functions are reciprocal, where the trivial case regards that cos(-θ)=cos(θ). Phase functions can be isotropic or anisotropic and we'll see them in detail in the further chapters. Another distinction is that for particles at microscopic level we generally use Raylegh scattering while for those at macroscopic level with use Mie theory (Henyey-Greenstein is a generalization of the latter).

Now getting back to our BSSRDF, the incoming radiance will decrease exponentially with distance s. This is referred to as the **reduced radiance**. which is the radiance that has entered the medium but has not yet scattered or been absorbed. And that's the first part into which the radiance is decomposed (the **diffuse radiance** $L_d(x, \vec{w})$ being the second term).

$$L_{ri}\left(x_i + s\,\vec{w}_i, \vec{w}_i\right) = e^{-\sigma_t s} L_i\left(x_i, \vec{w}_i\right)$$

for more on the topic of volume rendering, [see -> LightTransport->theory->1993 - (Direct) Volume Rendering]

Exponential Random Walks

Distance Sampling

$$T = 1 - e^{-\sigma s}$$

This gives back the actual ratio of uncollided particles to initial particles, having travelled *s* with absorption σ.
That corresponds to a change in radiance along the ray as light travels through a medium filled with absorbing particles.
Take care that here T(ransmittance) is a CDF, not a PDF.

In[102]:=
```
Clear[σ, s];
transmittance = 1 - Exp[-(σ * s)];

𝒟 = ProbabilityDistribution[{"CDF", transmittance}, {s, 0, ∞}]
D[transmittance, s](* or do it straightly with differentiation *)
```

Out[104]= ProbabilityDistribution$\left[\begin{cases} e^{-\text{x}\sigma}\sigma & \text{x} > 0 \\ 0 & \text{True} \end{cases}, \{\text{x}, 0, \infty\}\right]$

Out[105]= $e^{-s\sigma}\sigma$

This is the actual PDF, i.e. the normalization of an exponential distribution.

$$tPDF = \sigma e^{-\sigma s}$$
$$t = e^{-\sigma s}$$

this is the probability that a particle has an interaction in *ds* having travelled *s* into a medium σ.
This probability is a logarithmic relationship between the transmission of light and the product of the absorption in the medium and the distance travelled.

Now, if we lookup 'exponential distribution' in Wikipedia we'll see it defined as :
"The probability distribution of the time between events in a Poisson point process".
A Poisson point process is characterized via the Poisson distribution.
The Poisson distribution is the probability distribution of a random variable K such that the probability that K equals k is:

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Now what's the probability that something will happen between events.. zero (because that something happens.. is the event)

$$P(X=0) = \frac{\lambda^0 e^{-\lambda}}{0!} = e^{-\lambda}$$

How are our events ? Indipendent.
And if instead of dealing with the unit time we deal with a period, then

$$P(X=0) * P(X=0) * P(X=0) * \ldots = e^{(-\lambda t)}$$

which is our exponential distribution :)

Now our random walk is considered instantaneous, ain't taking any time at all, ie. is non-transient, but we can still use the same paradigm when we consider each indipendent particle collision as our event. Because here we're not interested in the events (which would be the scattering) themself but in the 'in-between events' (which is absorption).. but what is in-between each particle collision ?.. the distance (of light travelling in the matter) it will take to reach the next particle (collision event).

So what's telling us the PDF above as a general case ?.. the density of time until the next event will happen.
Effectively λ (in a Poisson distribution) is not a time duration but a rate of the unit of time, that's also why in an Exponential distribution, it is $1/\lambda$ which is the decay rate instead.

So what's between our events when not in a 'time' framework but in a 'physical/geometric' one, matter.
The primary causes of attenuation in matter are the photoelectric effect and the compton scattering.
How is matter characterized here ? With an extinction coefficent, aka the total cross section.

So we can say that $e^{-s\sigma}$ is the statistical density of the material expressed as transmission.
Why do we need it ? Because as we'll see below by inverting the integral of a density, we'll get a sample of the quantity that bounds that density, in this case we'll get a distance sample that characterize our light transmission so that, the sample corresponds (is proportional) to a certain density that needs to be considered if we wanna our sample to be unbiased.

When we do something like this : ξ=Ft(x) (where F is the above CDF and ξ is a uniform random variate ~ (0,1)), - what are we saying actually ? That the cumulative probability of the radiant flux to reach a distance s without incurring in a collision, is our random number. Or in other words, the probability that a certain distance will be travelled is the differential of the material density (as transmission) around that point or also that a certain distance travelled is proportional to the rate of change of the radiance flux in the material by the cross section (that's why we wanna sample a distance to simulate stochastically via a randomwalk the material transmittance).

$$\frac{d\Phi}{ds} = \frac{\Phi(s+ds) - \Phi(s)}{ds} = -\sigma\Phi \Rightarrow \Phi = e^{-\sigma s}$$

[see → https://en.wikipedia.org/wiki/Beer-Lambert_law for the derivation]
[see → https://www.sciencedirect.com/topics/mathematics/exponential-distribution for other insights]

[see → The Exponential Distribution and Its Applications , for a full complete treatment]

In[106]:=
```
Clear[f, s, σ];
f[s_, σ_] := Exp[-(s * σ)];

With[{σ = 0.5},
 Show[
  Plot[f[s, σ], {s, 0, 5}, PlotRange → All,
   PlotStyle → {Red, Opacity[0.75]}, AxesLabel → {"Distance", "Transmittance"},
   Filling → Axis, LabelStyle → Directive[RGBColor[0.5], Bold]],
  Plot[f[s, σ], {s, 1, 1.1}, Filling → 0, PlotRange → All]
 ]]
```

Out[108]=

I mean, the flux at (s+ds) minus the flux at s divided by ds is the reduced radiance, ie. the radiance multiplied by the material cross section, so that formally, the difference in radiant flux divided by a infinitesimal displacement *s* is the radiance around *s* and that, is (proportional to) the exponential decay that occurs within s - s+ds.

So let's see how to sample exponential distances by inverting the CDF of an exponential distributions.

We saw that the quantity describing the path length between scatterings can be considered a random variable being characterized by some probability density function defined for a defined interval such that :

$$\int_a^b p(\xi)\, d\xi = 1$$

Now we can simulate a random event for which the variable falls with frequency p(x)dx in the interval (x, x+dx) by choosing a random number R from a uniform distribution ϵ (0,1) and requiring :

$$R = \int_a^x p\,(\xi)\,d\xi$$

We know that integrating the interval from the lower-bound to our variable is effectively computing the CDF of the probability function till that point. All we're left to do is to solve for the variable of interest. This the Monte Carlo method.

In[109]:=
```
Assuming[{σ > 0}, Solve[ξ == 1 - ℯ^-(s σ), s]] (* 1-e^-(sσ) is the CDF of our exponential *)
```

Out[109]=
$$\left\{\left\{s \to \text{ConditionalExpression}\left[\frac{2\,i\,\pi\,C[1] + \text{Log}\left[\frac{1}{1-\xi}\right]}{\sigma}, C[1] \in \mathbb{Z}\right]\right\}\right\}$$

Where we get our sampling routine as

In[110]:=
```
-Log[1/(1-ξ)]/σ;
(* the (1/1-ξ) is still a random number ξ~U(0,1) so eventually we get *)
sampleDistance = -Log[1 - ξ]/σ;
(* while it looks like we save a substraction by using just the ξ..
  it's better to leave it as 1-ξ because ξ could turn out to be 0 and so
  we'd take the Log[0] which is undefined while it's not for Log[1] *)
```

We could have used also Mathematica build-ins

In[112]:=
```
Clear[σ, x]
PDF[ExponentialDistribution[σ], x]
CDF[ExponentialDistribution[σ], x]
InverseCDF[ExponentialDistribution[σ], ξ]
```

Out[113]=
$$\begin{cases} e^{-x\,\sigma}\,\sigma & x \geq 0 \\ 0 & \text{True} \end{cases}$$

Out[114]=
$$\begin{cases} 1 - e^{-x\,\sigma} & x \geq 0 \\ 0 & \text{True} \end{cases}$$

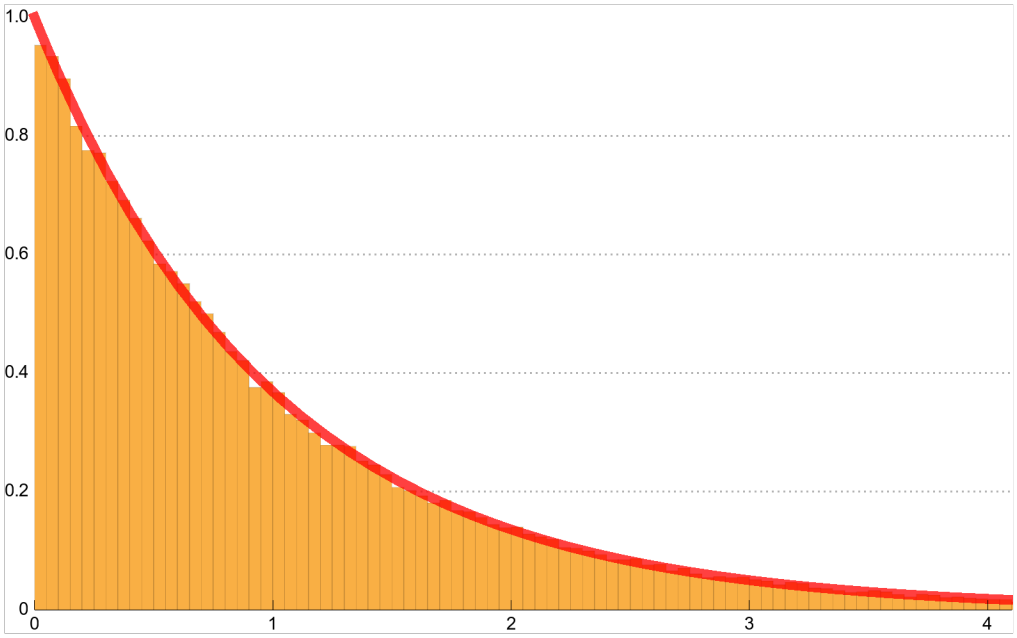
Out[115]=
$$\text{ConditionalExpression}\left[\begin{cases} -\frac{\text{Log}[1-\xi]}{\sigma} & \xi < 1 \\ \infty & \text{True} \end{cases}, 0 \leq \xi \leq 1\right]$$

$$\text{sampleExpDis} = -\frac{\text{Log}[1-\xi]}{\sigma}$$

Let's check this really works ..

In[116]:=
```
Clear[f, s, σ];
f[s_, σ_] := Exp[-(s * σ)];
sampleExpDis[σ_] := -Log[1 - #]/σ &[RandomReal[]]
With[{σ = 1.0},
 Show[
  Histogram[ParallelTable[sampleExpDis[σ], {i, Range[10^5]}],
   256, "PDF", PlotTheme → "Business"],
  Plot[f[s, σ], {s, 0, 5}, PlotStyle →
    Directive[Red, Thickness[0.01], Opacity[0.75]]]
 ]]
```

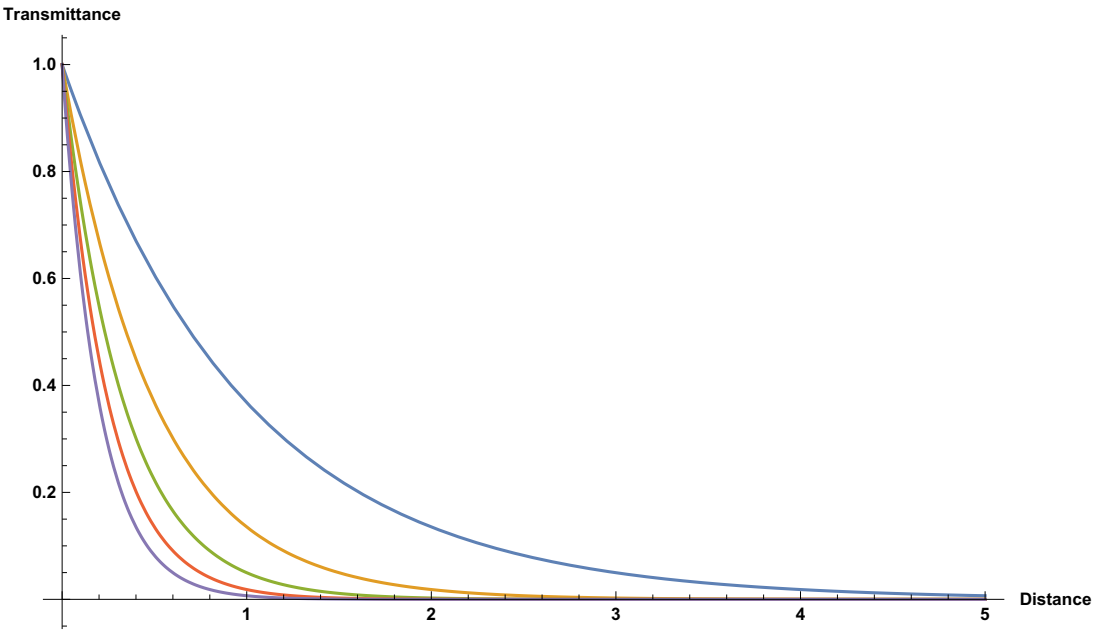
Out[119]=



Let’s plot some stuff ..

In[120]:=
```
Plot[Table[f[s, σ], {σ, 1, 5}] // Evaluate, {s, 0, 5},
 PlotRange → All, AxesLabel → {"Distance", "Transmittance"},
 LabelStyle → Directive[RGBColor[0.5], Bold]]
(*Plot[Evaluate[Table[f[s,σ],{σ,0,1}]],{s,0,10},PlotRange→All]*)
(*same as above*)
```

Out[120]=

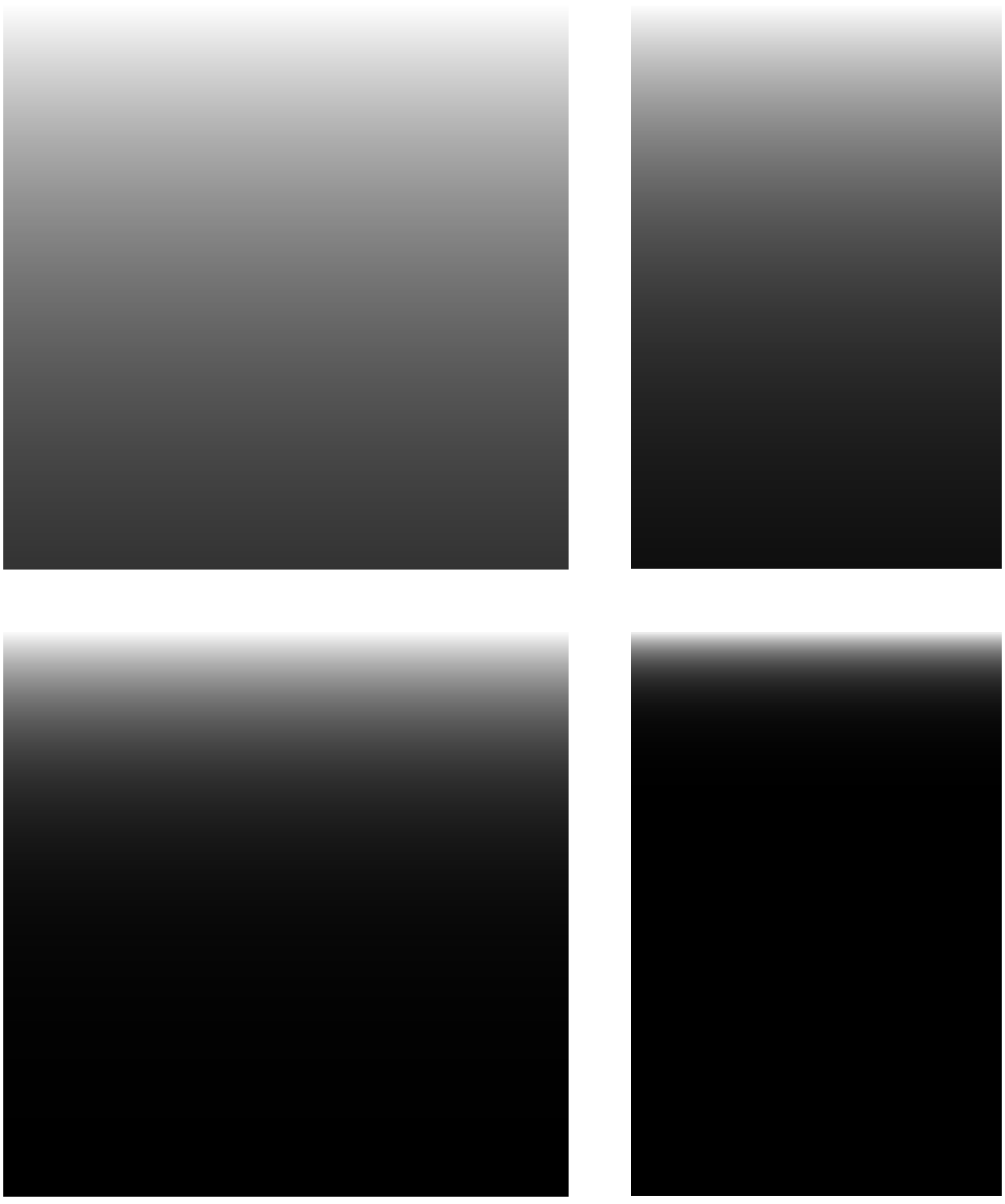


As we can see the more into the medium the less transmittance.
Also the higher the transmittance the less distance to travel for full absorption.

Analytically draw transmittance over the unit square

In[121]:=
```
(* here we just need the exponential fnc to feed it with the s-
 distance over the unit square, we divide the unit by the freq factor
 to get a smooth gradient and use a matrixplot to draw the gradient *)

ClearAll["Global`*"];
freq = 256;
fncScaling = False;
imageSize = Medium;

f[i_, σ_] := Module[{s, d, vSize},
  vSize = 20;
  s = (i / (freq)) * vSize; (* get step distance in 0,1 *)
  d = Exp[-(s * σ)];
  1 - d
 ]

GraphicsGrid[{
  {With[{σ = 0.08},
    expdata = Table[f[x, σ], {x, freq}, {y, freq}];
    (* yup, we could just eval one single column *)
    MatrixPlot[expdata, PlotRange → {0, 1},
     ColorFunctionScaling → fncScaling, Frame → False, ImageSize → imageSize]
   ], With[{σ = 0.14},
    expdata = Table[f[x, σ], {x, freq}, {y, freq}];
    MatrixPlot[expdata, PlotRange → {0, 1},
     ColorFunctionScaling → fncScaling, Frame → False, ImageSize → imageSize]
   ]},
  {With[{σ = 0.32},
    expdata = Table[f[x, σ], {x, freq}, {y, freq}];
    MatrixPlot[expdata, PlotRange → {0, 1},
     ColorFunctionScaling → fncScaling, Frame → False, ImageSize → imageSize]
   ], With[{σ = 1},
    expdata = Table[f[x, σ], {x, freq}, {y, freq}];
    MatrixPlot[expdata, PlotRange → {0, 1},
     ColorFunctionScaling → fncScaling, Frame → False, ImageSize → imageSize]
   ]}}]
```

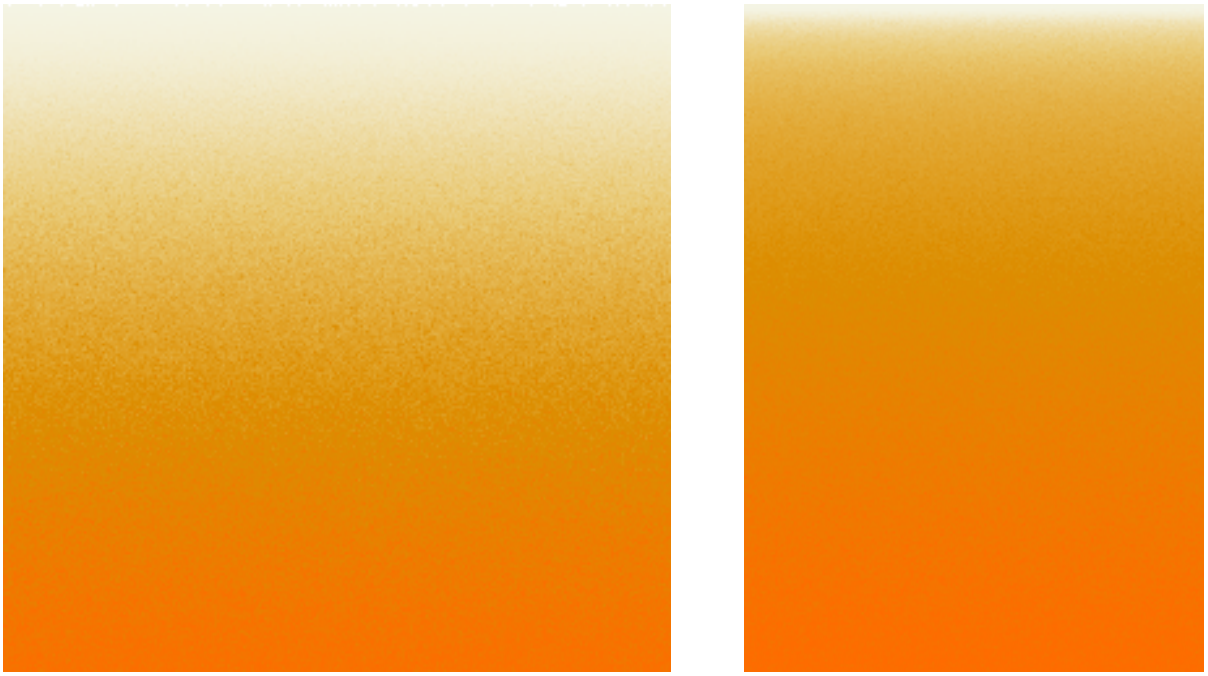
Out[126]=



Stochastically, draw a transmittance gradient with MC

In[127]:=
```
(* here instead we need everything we got till now, the exp fnc,
its PDF and its inverted CDF, we ain't going to simulate the exp
 gradient with a randomwalk however we'll get the sdistance from the
 stochastic evaluation of the inverse CDF over the unit square *)

ClearAll["Global`*"];
imageSize = Medium;
fncScaling = True;
(* set it to False to have the gradient properly in BW *)
nRndSamples = 64;      (* the higher, the less noise in the gradient,
the higher calc time, smooth for >512 *)
freq = 256;            (* because we divide our rnd by this,
it will also influence the stochastic outcome *)

expFnc[σ_, s_] = Exp[-(σ * s)];
expPDF[σ_, s_] = σ * Exp[-(σ * s)];
expSample[σ_, rnd_] = -Log[1 - rnd] / σ;

f[i_, σ_] := Module[{s, d, datas},
  datas = Table[expSample[σ, RandomReal[] / i], {x, nRndSamples}];
                       (* draw n 'probable' distances; note: rnd/freq *)
  s = Mean[datas];      (* get their average *)
  s = s / expPDF[σ, s];   (* divide it by the PDF, aka importance sampling *)
  s = s * 20;           (* optional: compress/enlarge exp over the unit *)
  d = expFnc[σ, s];     (* feed the exp fnc with the stochastic distance *)
  d                     (* get back with our transmittance at the point *)
 ]

GraphicsGrid[{
  {With[
    {σ = 0.1},                                          (* different sigmas *)
    expdata = ParallelTable[f[x, σ], {x, freq}, {y, freq}];
    MatrixPlot[expdata, PlotRange → {0, 1},
     ColorFunctionScaling → fncScaling, Frame → False, ImageSize → imageSize]
   ],
   With[{σ = 1},
    expdata = ParallelTable[f[x, σ], {x, freq}, {y, freq}];
    (* note, parallel execution *)
    MatrixPlot[expdata, PlotRange → {0, 1}, ColorFunctionScaling → fncScaling,
     Frame → False, ImageSize → imageSize]
   ]}
 }]
```

Out[136]=



Eventually the sampling stuff we got from this chapter :

$$\text{transmittancePDF} = \sigma e^{-\sigma s}$$

$$\text{sampleDistance} = -\frac{\text{Log}[1-\xi]}{\sigma}$$

==========

Direction Sampling

[see -> PT SSS Using Anisotropic Phase Functions and Non-Exponential Free Flights]

The relationship between incoming and outgoing directions is represented by a phase function, which **assigns a probability to each incoming/outgoing direction pair**. Since only the relative angle between the two directions is relevant, **phase functions map one-dimensional angle quantity to a one-dimensional probability**.

The most popular phase function is Henyey-Greenstein, often abbreviated as HG. The parameter g (generally called the mean cosine) controls the anisotropic bias, with g = 0 being an isotropic distribution, g = 1 being a forward Dirac response and g = –1 being a backward Dirac response. Here 'Dirac' means close to a single direction, ie. the lobe is so elongated that looks like a sharp sword... ie. at g=1, μ=1; at g=-1, μ=-1;

$$P(\Theta) = \frac{1}{4\pi} \frac{1 - g^2}{\left(1 + g^2 - 2\,g\,\text{Cos}[\Theta]\right)^{3/2}},$$

$$\text{such that} \int_0^{\pi} P(\Theta)\, 2\pi\, \text{Sin}[\Theta]\, d\Theta = 1$$

$$\text{with } g = \int_0^{\pi} p(\Theta)\, \text{Cos}[\Theta]\, 2\pi\, \text{Sin}[\Theta]\, d\Theta$$

It is common practice to express the Henyey – Greenstein function as the function $p(\cos\Theta)$ **;

$$P(\mu) = \frac{1}{4\pi} \frac{1 - g^2}{\left(1 + g^2 - 2\,g\,\mu\right)^{3/2}},$$

$$\text{such that} \int_0^{\pi} P(\mu)\, \text{Cos}[\Theta]\, 2\pi\, \text{Sin}[\Theta]\, d\Theta = 1$$

$$\text{with } g = \int_0^{\pi} p(\mu)\, \text{Cos}[\Theta]\, d\mu$$

** BTW, **why Cos[θ]** ? Because when we try to find out what component of one vector is in the same direction as another vector, we do something called *projection* of the first vector onto the second one, **that will give us the fraction of the length of a vector pointing in the same direction as the other vector (which is the definition of dot product)**. To do this, draw a line from the tip of the first vector down onto the second vector. This line should be perpendicular to the second vector. When we have done this, we have literally formed a right-angled triangle whose hypotenuse is the first vector, and whose adjacent side is the projection of that vector onto the second vector. **So the ratio of lengths** (projection of vector 1 onto vector 2) / (vector 1) is literally given by (adjacent side of right triangle) / (hypotenuse of right triangle), and this ratio **is by definition the cosine of the angle**. That's what cosine means. It's the ratio of the lengths of those two particular sides of the right triangle.

With **isotropic** we simply mean that a particle scatters photons in every direction with equal probabilities, so its PDF will be 1/2.

In[137]:=
```
normfactor = 2 π; (1/(4 π) * 1) * normfactor
```

Out[137]=
$$\frac{1}{2}$$

Effectively for g=0 the whole second term in the HG equation will be simply 1 reducing HG PDF to that of a sphere of directions.
When g≠0 then it is **anisotropic.**

[see -> https://www.scratchapixel.com/lessons/advanced-rendering/volume-rendering-for-artists]
Now, how does it happen that when a photon collides with an atomic particle the direction it will take is biased toward a certain direction ? Simply put, it does not happen :)

Effectively when a photon collides with a single particle its chances of taking a certain direction are

_always _ uniformly distributed. **It is when we have more than one particle in the medium that we may have anisotropy. This happens because of the so called wave interference**. In the case of dense homogeneous media the spacing between particles may be much less than the light wavelength and in that particular case photons traveling through the medium are affected by what is called the phenomenon of wave interference. There may be a constructive or desctructive interference. These interferences depends on the phase shift between the two waves. In mediums made of organised matter, that is, matter where particles are so close to each other that they arrange themselves in organised structures (aka regular patterns), light waves are heavily affected by the wave interference phenomenon while they propagate through these mediums.

However remeber that we said that here we'll deal with linear transport theory, - particles do not interact with each other nor electromagnetic waves collides with each others. Beside the involved theory, the simulation will take ages to converge. That's the cool thing about the HG phase function. It can be used as a mathematical tool to describe anisotropy without having to simulate its underlying physical behaviour.

The anisotropy or in this case the asymmetry factor *g* depends (for the same reason above) on the size of the particles. Particles that are small compared to the wavelength of the radiation, such as air molecules, have asymmetry factors close to zero. Larger particles, such as cloud droplets, typically have asymmetry factors ~0.85 for visible radiation, consistent with strong forward scattering. **g** is called the mean cosine and is equal to :

$$g = \langle \text{Cos}\Theta \rangle = \frac{\int p(u)\, \cos(u)\, du}{\int p(u)\, du}$$

Why ‘mean’ ? Because it takes into account all the scattering bodies of the medium as a whole ‘mean’ scattering body.

As before, having a probability distribution functions we need a sampling fnc from that to be able to use in a MC env.
Let’s start by seeing if its normalized and from there get it CDF and eventually invert it.

In[1]:=
```
ClearAll["Global`*"]
pHG[u_, g_] = 1/(4 π) * (1 - g^2)/(1 + g^2 - 2 g u)^(3/2); (* u=Cos[Θ] where Θ ∈ [0,π]rad*)
Integrate[pHG[u, g], {u, -1, 1}, Assumptions → g > -1 && g < 1]
(* let's check it's a PDF.. integrates to 1 *)

(* normalize it so it becomes a PDF *)
𝒟 = ProbabilityDistribution[pHG[u, g],
  {u, -1, 1}, Assumptions → g > -1 && g < 1, Method → "Normalize"]
𝒟 /. ProbabilityDistribution → Integrate (* re-check it integrates to 1 *)

(* from PDF find its CDF *)
CDF[𝒟]
(*tmpCDF=Integrate[2 Pi pHG[u,g],{u,-1,u},Assumptions→g>-1&&g<1&&u<1,
  GenerateConditions->False]*)(* same as above *)

(* eventually invert it to find the direction sampling *)
InverseCDF[𝒟, ξ];
FullSimplify[%]
(*FullSimplify[Solve[tmpCDF==ξ,u]] *)(* same as above *)
```

Out[3]=
$$\frac{1}{2\pi}$$

Out[4]=
$$\text{ProbabilityDistribution}\left[\frac{1-g^2}{2\left(1-2\,\dot{x}\,g+g^2\right)^{3/2}},\ \{\dot{x},\ -1,\ 1\},\ \text{Assumptions}\to g>-1\ \&\&\ g<1\right]$$

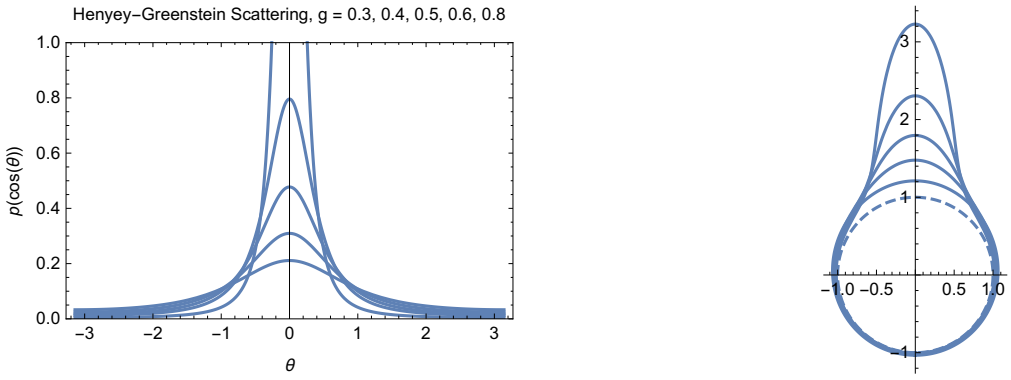
Out[5]=
$$1$$

Out[6]=
$$\text{Function}\left[\dot{x},\ \begin{cases} 1 & \dot{x}\geq 1 \\ \dfrac{(-1+g)\left(-1-g+\sqrt{1-2\,\dot{x}\,g+g^2}\right)}{2\,g\,\sqrt{1-2\,\dot{x}\,g+g^2}} & -1<\dot{x}<1 \\ 0 & \text{True} \end{cases},\ \text{Listable}\right]$$

Out[8]=
$$\text{ConditionalExpression}\left[\begin{cases} -1 & \xi\leq 0 \\ 1 & \xi\geq 1 \\ -\dfrac{(-1+g)^2+2\,(-1+g)\,\left(1+g^2\right)\,\xi-2\,g\,\left(1+g^2\right)\,\xi^2}{(1+g\,(-1+2\,\xi))^2} & \text{True} \end{cases},\ 0\leq\xi\leq 1\right]$$

```
(* borrowed from → dEon : A Hitchhiker's Guide to Multiple Scattering *)
GraphicsRow[{
  Show[
   Plot[pHG[Cos[t], .8], {t, -Pi, Pi}, PlotRange → {0, 1}],
   Plot[pHG[Cos[t], .6], {t, -Pi, Pi}, PlotRange → All],
   Plot[pHG[Cos[t], .5], {t, -Pi, Pi}, PlotRange → All],
   Plot[pHG[Cos[t], .4], {t, -Pi, Pi}, PlotRange → All],
   Plot[pHG[Cos[t], .3], {t, -Pi, Pi}, PlotRange → All],
   Frame → True,
   ImageSize → 400,
   FrameLabel → {{p[Cos[θ]],},
     {θ, "Henyey-Greenstein Scattering, g = 0.3, 0.4, 0.5, 0.6, 0.8"}}],

  Show[
   ParametricPlot[{Sin[t], Cos[t]} (1),
    {t, -Pi, Pi}, PlotRange → All, PlotStyle → Dashed],
   ParametricPlot[{Sin[t], Cos[t]} (1 + pHG[Cos[t], 0.75]),
    {t, -Pi, Pi}, PlotRange → All],
   ParametricPlot[{Sin[t], Cos[t]} (1 + pHG[Cos[t], 0.68]),
    {t, -Pi, Pi}, PlotRange → All],
   ParametricPlot[{Sin[t], Cos[t]} (1 + pHG[Cos[t], 0.6]),
    {t, -Pi, Pi}, PlotRange → All],
   ParametricPlot[{Sin[t], Cos[t]} (1 + pHG[Cos[t], 0.5]),
    {t, -Pi, Pi}, PlotRange → All],
   ParametricPlot[{Sin[t], Cos[t]} (1 + pHG[Cos[t], 0.3]),
    {t, -Pi, Pi}, PlotRange → All]
  ]}, ImageSize → Full]
```

Out[107]=



So our HG stuff is :

In[9]:=
$$\text{pdfHG}[u\_, g\_] := \frac{1}{2}\,\frac{1-g^2}{\left(1+g^2-2\,g\,u\right)^{3/2}};$$

$$\text{cdfHG}[u\_, g\_] := \frac{(-1+g)\left(-1-g+\sqrt{1+g^2-2\,g\,u}\right)}{2\,g\,\sqrt{1+g^2-2\,g\,u}};$$

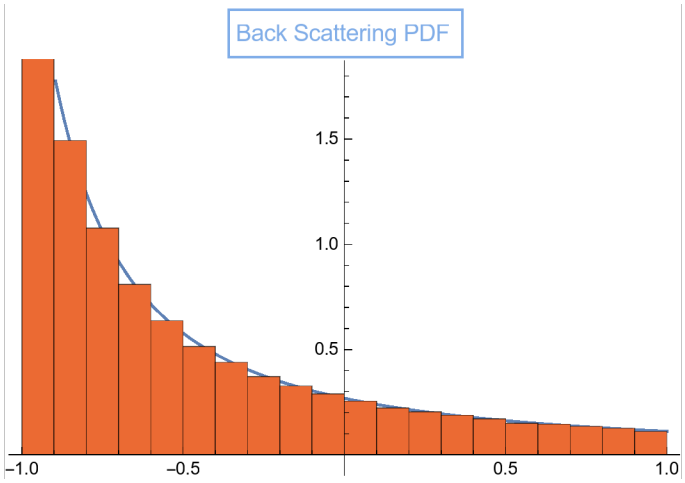
$$\text{sampleHG} = -\frac{(-1+g)^2+2\,(-1+g)\left(1+g^2\right)\xi-2\,g\left(1+g^2\right)\xi^2}{\left(1+g\left(-1+2\,\xi\right)\right)^2};$$

$$\text{samplingHG}[g\_] := -\frac{(-1+g)^2+2\,(-1+g)\left(1+g^2\right)\#-2\,g\left(1+g^2\right)\#^2}{\left(1+g\left(-1+2\,\#\right)\right)^2}\,\&[\text{RandomReal}[]];$$

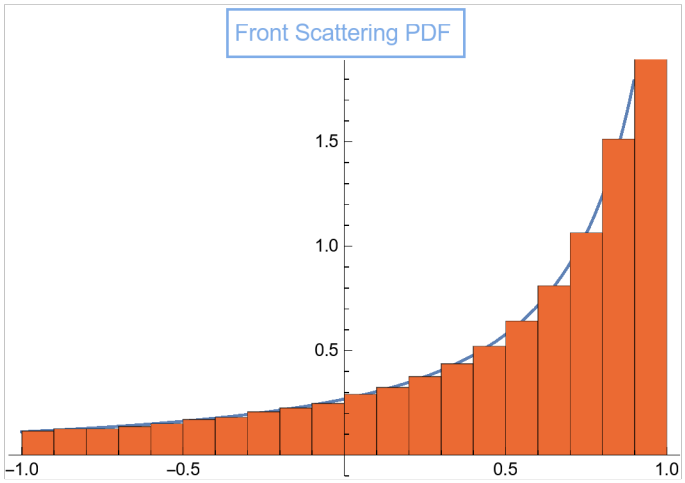
Now let's check we got something that works

In[13]:=
```
With[{g = -0.5}, (* back scattering *)
 Show[
  Plot[pdfHG[u, g], {u, -1, 1}, PlotLabel →
    Style[Framed["Back Scattering PDF"], 12, RGBColor[0.5, 0.68, 0.91]]],
  Histogram[ParallelTable[samplingHG[g], {i, Range[10^5]}],
   Automatic, "PDF", PlotTheme → "WarmColor", PlotRange → All]
 ]]
With[{g = 0.5}, (* front scattering *)
 Show[
  Plot[pdfHG[u, g], {u, -1, 1}, PlotLabel →
    Style[Framed["Front Scattering PDF"], 12, RGBColor[0.5, 0.68, 0.91]]],
  Histogram[ParallelTable[samplingHG[g], {i, Range[10^5]}],
   Automatic, "PDF", PlotTheme → "WarmColor", PlotRange → All]
 ]]
With[{g = 0.0}, (* isotropic *)
 Show[
  Plot[pdfHG[u, g], {u, -1, 1}, PlotLabel →
    Style[Framed["Isotropic scattering PDF"], 12, RGBColor[0.5, 0.68, 0.91]]],
  Histogram[ParallelTable[samplingHG[g], {i, Range[10^5]}],
   Automatic, "PDF", PlotTheme → "WarmColor", PlotRange → All]
 ]]
```

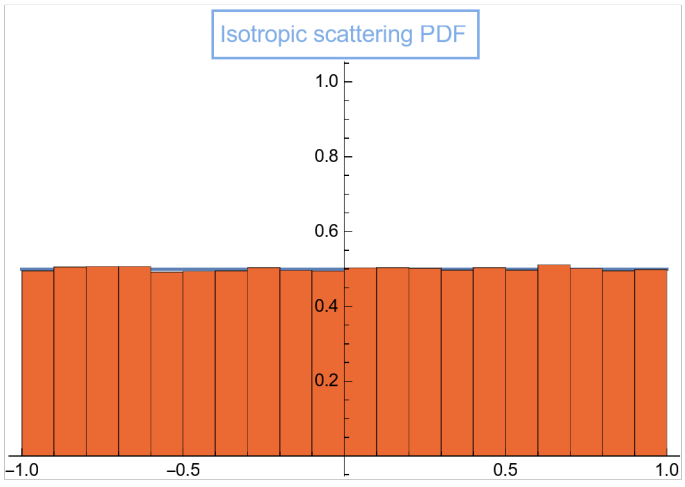
Out[13]=



Out[14]=



Out[15]=

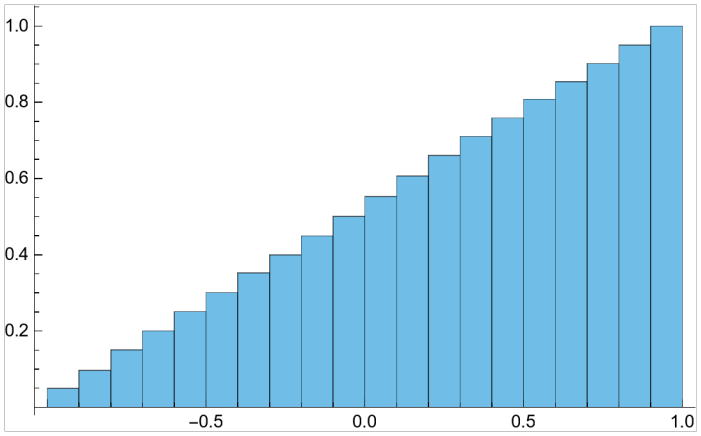


We see that for back scattering for example we have much more chances to get a cosine of -1 than anything else.

While for isotropic scattering we have a uniform distribution : 1/(b-a)=1/(1-(-1)=1/2=0.5.

In[16]:=
```
With[{g = 0.0}, (* visualize we're accumulating probabilities up to 1 *)
 Histogram[ParallelTable[samplingHG[g], {i, Range[10^4]}],
  Automatic, "CDF", PlotTheme → "PastelColor", PlotRange → All]]
```

Out[16]=



Not last, we can also check what PDF we get when explicitly use the cosine for an actual random variate.

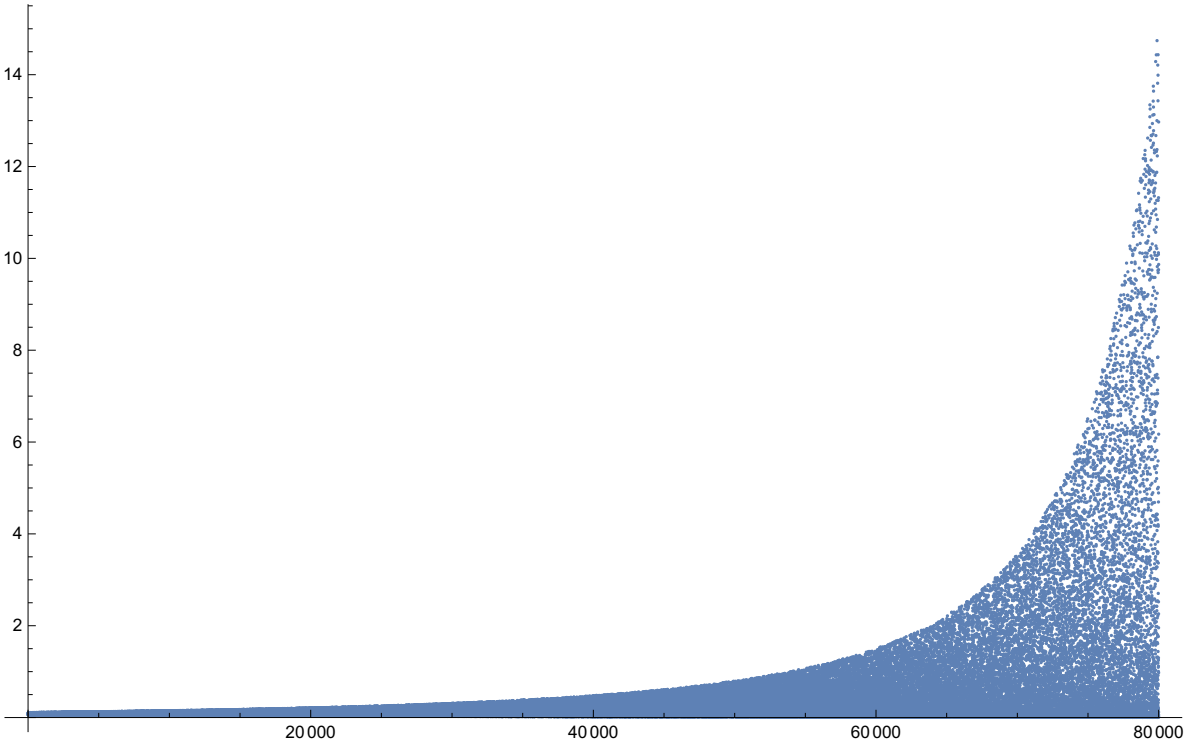
In[17]:=
```
Clear[g, ξ];
FullSimplify[pHG[sampleHG, g], Assumptions → (-1 < g < 1) && (0 < ξ < 1)]
```

Out[18]=
$$\frac{\left(1+g\left(-1+2\,\xi\right)\right)^3}{4\left(-1+g^2\right)^2\pi}$$

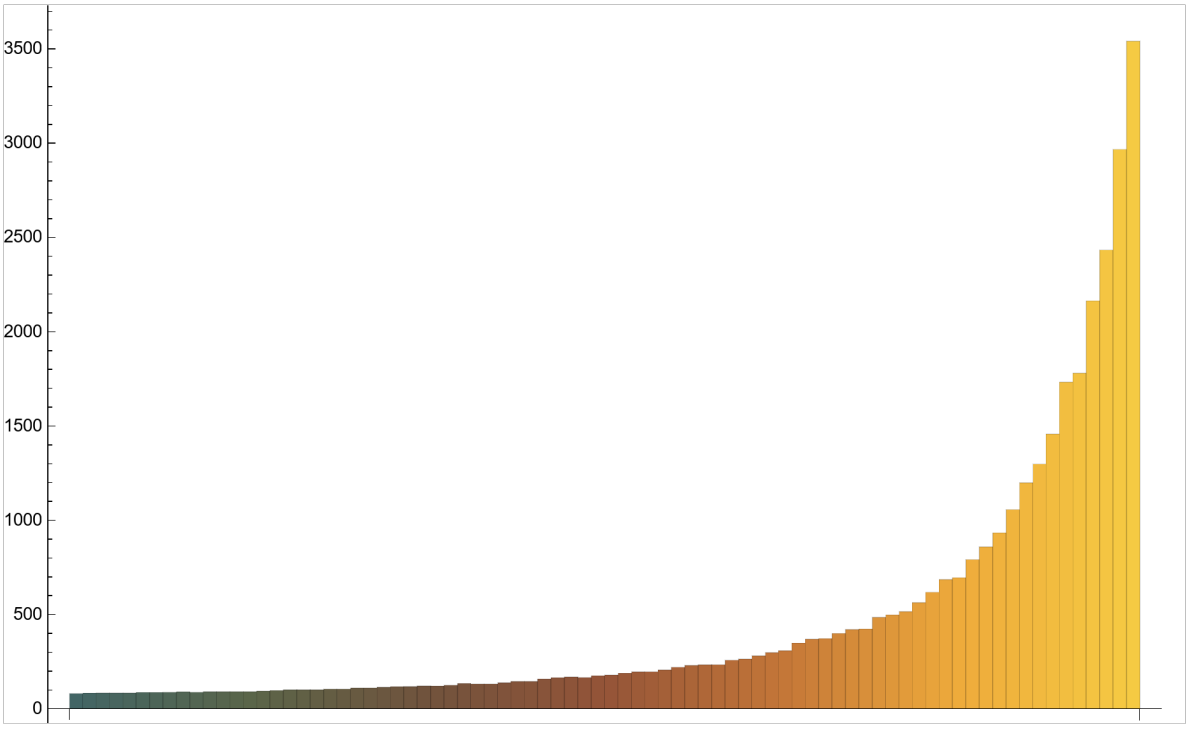
We can also check what's the full range of the PDF over our random variable, that of course will resemble the orange histograms above. We see we have some constant probability around g=0 and that starts increasing more and more while approach g=1;

In[19]:=
```
(* subdivide costheta range by .. *)
hgList = Table[pHG[samplingHG[g], g], {g, 0.1, 0.9, 0.00001}];
ListPlot[%, PlotRange → Full]

(* wanna a custom 'histogram' ? *)
plst = Partition[hgList, 1000];              (*divide list*)
slst = Map[Total, plst];                     (*get each chunck sum*)
BarChart[slst, ChartStyle → "FallColors"]    (*display them as bar charts*)
(* here it's nice to see that the green coloring is for constant prob *)
```

Out[20]=



Out[23]=



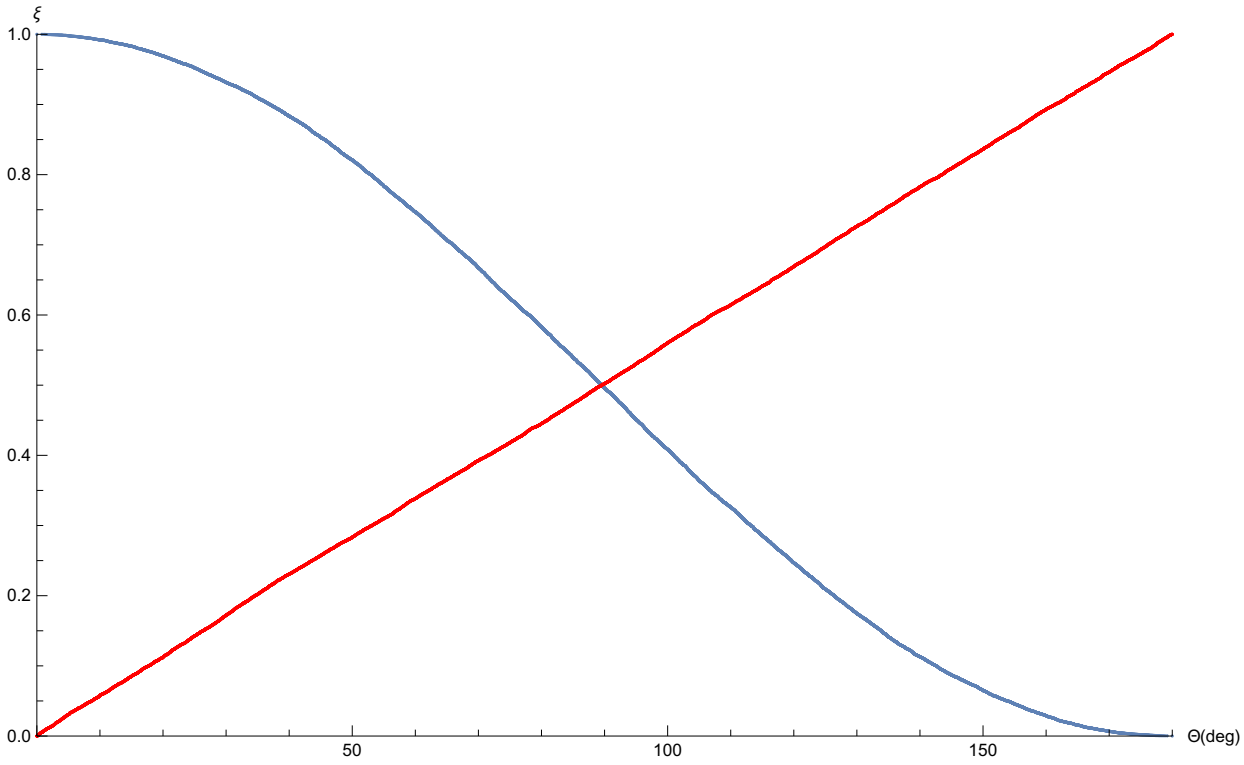
Following will rig this; ie. the problem here is to solve the below eq to obtain an analytical solution expressing the angle θ as a function of a uniformly distributed random variable $\xi \sim U(0,1)$:

$$\int_0^{\theta} P_{hg}(\theta')\, d\theta' = \xi$$

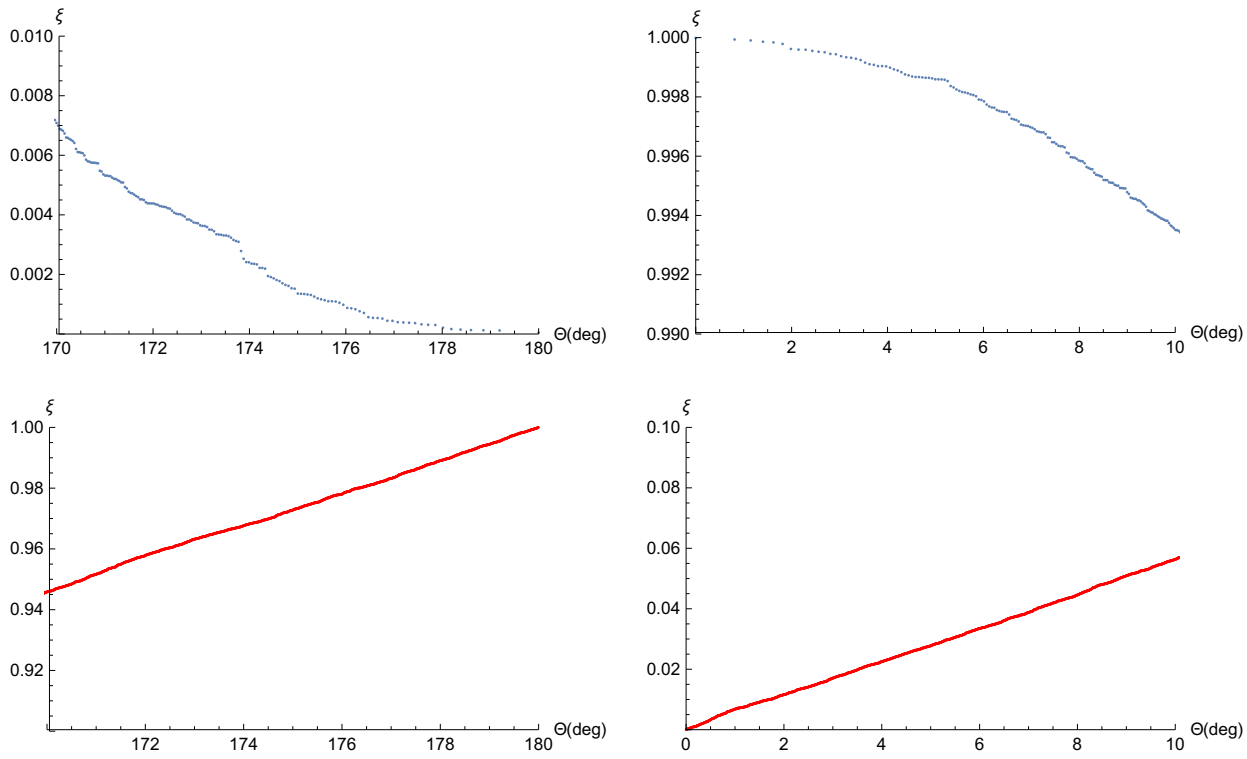
In[97]:=
```
rndTheta[isTheta_: False, g_: 0.0, step_: 0.001] :=
 Module[{ru, rhg, r, ttt, nPts, posHG, pttt, ttx, rndx, xcos, xyttt},
  If[isTheta,
   ru = Range[0, π, step];                              (* quadrature setup*)
   rhg = Rescale[pHG[ru, g]];
   (* these are probabilities(0-1) corresponding to those dθ*)
   ,
   ru = Range[-1, 1, step];
   rhg = Rescale[pHG[ru, g]];
  ];

  r = Sort[Table[RandomReal[], Length[rhg]]];
  (* let's shot n rnd variates as many interpolation pts *)
  ttt = Transpose[{r, ru, rhg}];

  nPts = Flatten[Sort[Nearest[rhg, r, 1]]];
  (* let's match those variates with their probabilities *)
  posHG = Flatten[Position[rhg, Alternatives @@ nPts]];
  (* list of pos that corresponds to pHG*)

  pttt = ttt[[posHG]];
  (* find out what entries correspond to those positions *)
  ttx = Transpose[pttt];
  rndx = ttx[[1]] ;
  (* and so the associated random variates, r *)
  xcos = ttx[[2]] ;
  (* with their associated cosines, ru *)

  If[isTheta,
   xyttt = Transpose[{xcos * (180 / π), rndx}] ;
   (*rearrange so x is theta(deg) and y is rnd*)
   ,
   xyttt = Transpose[{ArcCos[xcos] * (180 / π), rndx}];
  ];
  xyttt
 ]

(* plot ta shit *)
Show[
 ListPlot[rndTheta[False, 0.0, 0.0001],
  AxesLabel → {"Θ(deg)", "ξ"}, PlotRange → {{0, 180}, {0, 1}}],
 ListPlot[rndTheta[True, 0.0, 0.0002], PlotStyle → {Red}], ImageSize → Full
```

```
]
GraphicsGrid[{
  {
   ListPlot[rndTheta[False, 0.0, 0.0001],
    AxesLabel → {"Θ(deg)", "ξ"}, PlotRange → {{170, 180}, {0, 0.01}}],
   ListPlot[rndTheta[False, 0.0, 0.0001], AxesLabel → {"Θ(deg)", "ξ"},
    PlotRange → {{0, 10}, {0.99, 1}}]
  }, {
   ListPlot[rndTheta[True, 0.0, 0.0001], AxesLabel → {"Θ(deg)", "ξ"},
    PlotRange → {{170, 180}, {0.9, 1.0}}, PlotStyle → {Red}],
   ListPlot[rndTheta[True, 0.0, 0.0001], AxesLabel → {"Θ(deg)", "ξ"},
    PlotRange → {{0, 10}, {0.0, 0.1}}, PlotStyle → {Red}]}
 }, ImageSize →
  Full]
```

Out[98]=



Out[99]=



[see → The Use of the HGF in MC Simulations in Biomedical Optics]

The integral above has been numerically evaluated by using a fixed quadrature. This gives a monotonically increasing table of values as a function of θ. The obtained values have been normalized from 0 to 1. Then we shot as many random variates as many values we have. What are we doing ? Scattering probabilities have been uniformly discretized and we're now associating uniformly distributed random numbers to those values. Once got the scattering probability associated with a random number from there we'll get the scattering angle beloging to that probability. By construction, θ obtained by generating a set of uniformly distributed values ξ follow the distribution law given by the HG equation.

It clearly shows how costheta is not evenly distributed for small and big directions (blue graphs above) while theta is (red graphs).
Take care here we're evaluating thetas and costhetas for g=0 so all directions should be evenly distributed.
Remember that costheta approach belongs to MC context while the theta one is used in i.e. wave optics.

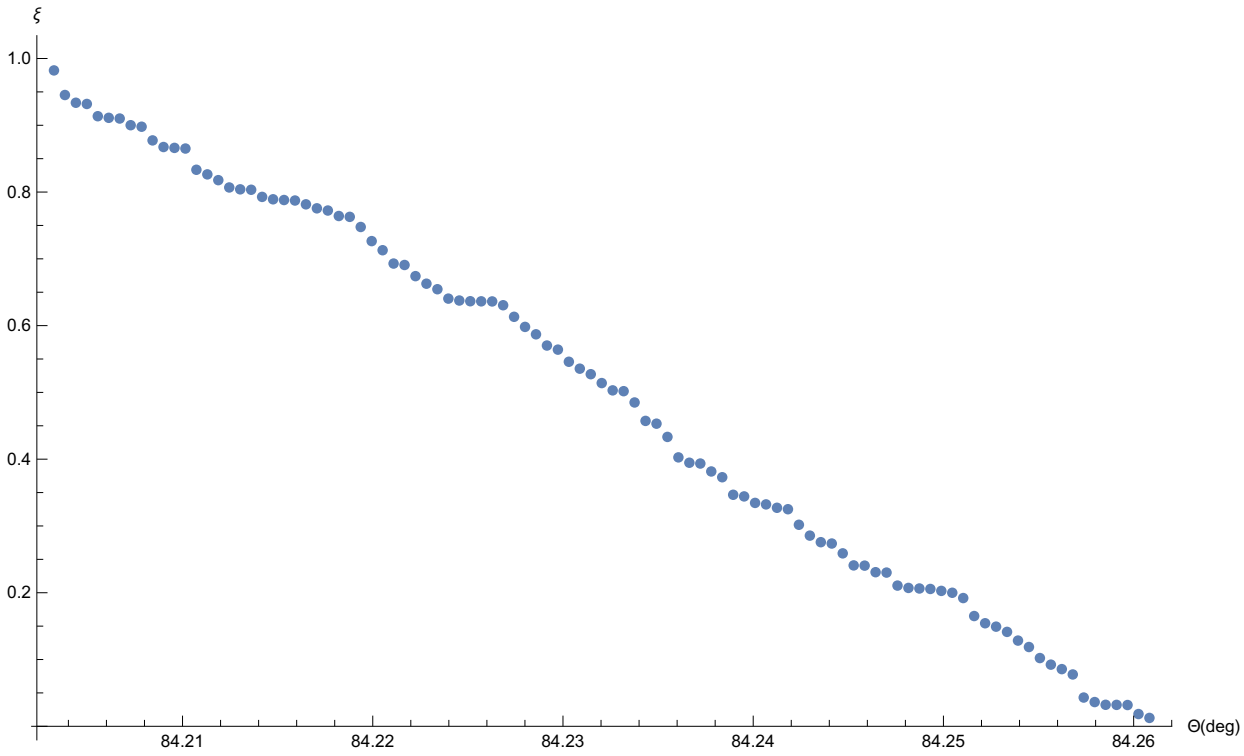
```
(*(* TODO :: let's try Toublanc (19996)
  method too where instead of normalizing in 0-1,
he's making all add up to 1 and compresses the random variable
 in that range while still uniformly distributed*)
phPDFs/Total[phPDFs];
Total[%];*)
```

This is also the proof of the inversion/transformation CDF method to get a sampling fnc we used

for both distance sampling and direction sampling. In the specific we can just integrate around x,x+dx and see if we have uniformly distributed directions. We need them in that fashion because for the law of large numbers, the more we get and the more their average/PDF, will resemble the expected outcome. That's what we do for example in the stochastic transmittance gradient drawing in the distance sampling chapter.

In[128]:=
```
rndCosTheta[step_: 0.001, costhetaa_: -1, costhetab_: 1, g_: 0.0] :=
 Module[{ru, rhg, r, ttt, nPts, posHG, pttt, ttx, rndx, xcos, xyttt},
  ru = Range[costhetaa, costhetab, step];              (* quadrature setup*)
  rhg = Rescale[pHG[ru, g]];
  (* these are probabilities(0-1) corresponding to those ∂θ*)

  r = Sort[Table[RandomReal[], Length[rhg]]];
  (* let's shot n rnd variates as many interpolation pts *)
  ttt = Transpose[{r, ru, rhg}];

  nPts = Flatten[Sort[Nearest[rhg, r, 1]]];
  (* let's match those variates with their probabilities *)
  posHG = Flatten[Position[rhg, Alternatives @@ nPts]];
  (* list of pos that corresponds to pHG*)

  pttt = ttt[[posHG]];
  (* find out what entries correspond to those positions *)
  ttx = Transpose[pttt];
  rndx = ttx[[1]] ;
  (* and so the associated random variates, r *)
  xcos = ttx[[2]] ;
  (* with their associated cosines, ru *)

  xyttt = Transpose[{ArcCos[xcos] * (180 / π), rndx}];
  xyttt
 ]
ListPlot[rndCosTheta[0.00001, 0.1, 0.101],
 AxesLabel → {"Θ(deg)", "ξ"}, PlotRange → Automatic, ImageSize → Full]
```

Out[129]=

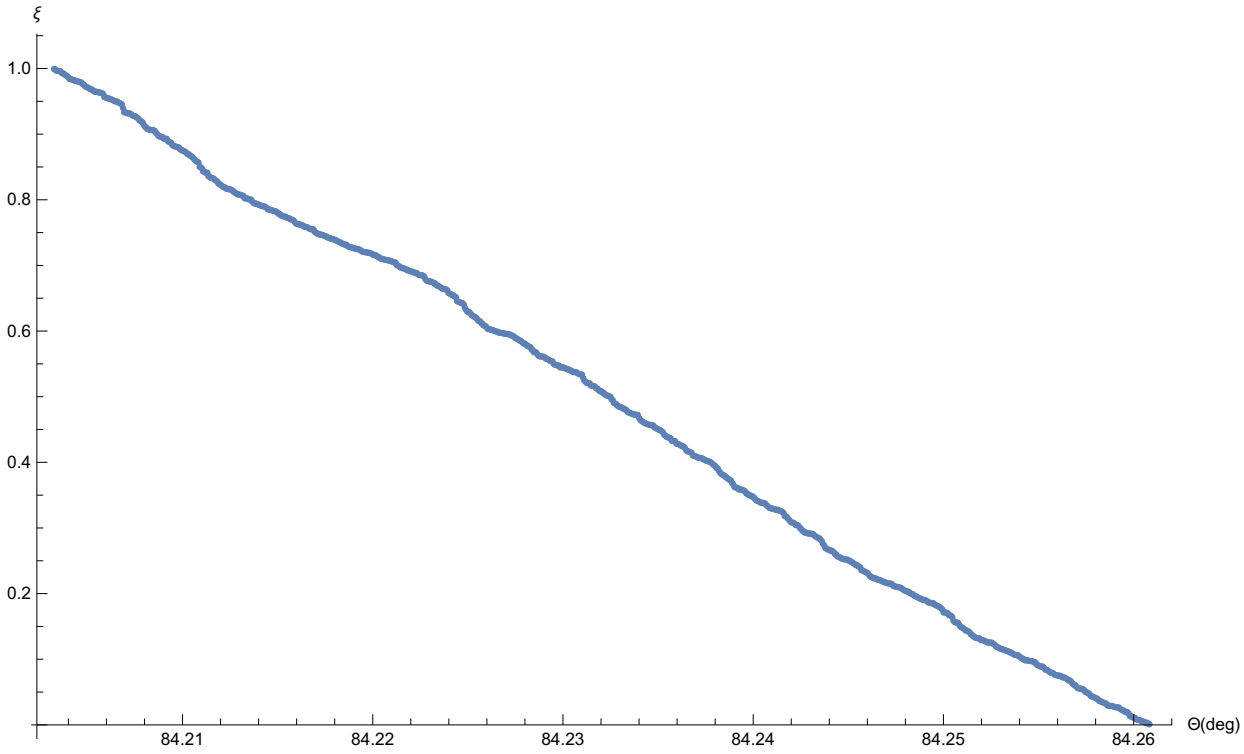


Above for example we're subdiving the interval θ - θ+$d\theta$ where $d\theta$ is 0.001 with 100 subdivisions (0.001/0.00001) and we clearly see that all the directions we got are uniformly distributed, that means they'll work cool in a MC context where we shot uniformly distributed random numbers to get those angles in our simulation. Try to repeat the eval with the same param and you'll see we get everytime different directions but still all nicely evenly distributed.

Btw, just lower the *step* parameter (so more subd) to have a nicer plot (where however we won't see anymore the single points).

In[130]:=
```
ListPlot[rndCosTheta[0.000001, 0.1, 0.101],
 AxesLabel → {"Θ(deg)", "ξ"}, PlotRange → Automatic, ImageSize → Full]
```

Out[130]=



Eventually, the sampling stuff we got from this chapter :

$$\text{Cos}[\theta] = \begin{cases} -\dfrac{(-1+g)^2+2\,(-1+g)\,\left(1+g^2\right)\,\xi-2\,g\,\left(1+g^2\right)\,\xi^2}{(1+g\,(-1+2\,\xi))^2} & g \neq 0 \\ 2\,\xi-1 & g = 0 \end{cases}$$

We'll see in a bit why here we don't have to account for the PDF of the directions we sample.
Btw, when we have a piecewise it generally means in real code we'll have an *if* statement to just switch to the pertinent routine.

==========