Mathematica Notebook for the SIGGRAPH 2013 talk "Background: Physics and Math of Shading"

This notebook contains some computations referenced in the course notes.

```
Off[NIntegrate::inumr]
SetOptions[Plot, PlotRange → {0, All}];

(*Based on the default Matlab color scheme, with some tweaks*)
pCol = RGBColor[0, 0, 1]; (*blue*)
bCol = RGBColor[0.85, 0.0, .85]; (*magenta*)
tsCol = RGBColor[0, 0.75, 0.75]; (*cyan*)
trCol = RGBColor[1, 0, 0]; (*red*)
abcCol = RGBColor[0, 0.75, 0]; (*green*)
sgdCol = RGBColor[1, 0.75, 0]; (*orange*)
gtrCol = RGBColor[0.6, 0.3, 0.0]; (*brown*)
```

Phong NDF

This is the unnormalized Phong distribution function:

In[10]:=
```
unnormalizedphong = Cos[θ]^αp;
```

Compute the normalization factor (relative to projected area) for the Phong distribution function:

In[11]:=
```
phongnormf = Integrate[unnormalizedphong Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αp > 0}]
```

Out[11]= $\frac{2\pi}{2+\alpha p}$

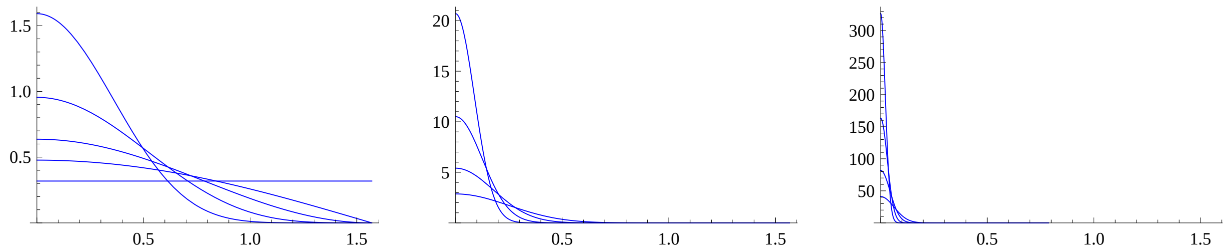
In[12]:=
```
phong = unnormalizedphong / phongnormf
```

Out[12]= $\frac{(2+\alpha p)\,\mathrm{Cos}[\theta]^{\alpha p}}{2\pi}$

Here are the distribution curves for some logarithmically spaced cosine powers (as well as 0, which corresponds to the uniform distribution):

In[13]:=
```
GraphicsRow[{Plot[phong /. αp → # & /@ {0, 1, 2, 4, 8}, {θ, 0, π / 2}, PlotStyle → pCol],
  Plot[phong /. αp → # & /@ {16, 32, 64, 128}, {θ, 0, π / 2}, PlotStyle → pCol],
  Plot[phong /. αp → # & /@ {256, 512, 1024, 2048},
   {θ, 0, π / 2}, PlotStyle → pCol]}, ImageSize → Full]
```

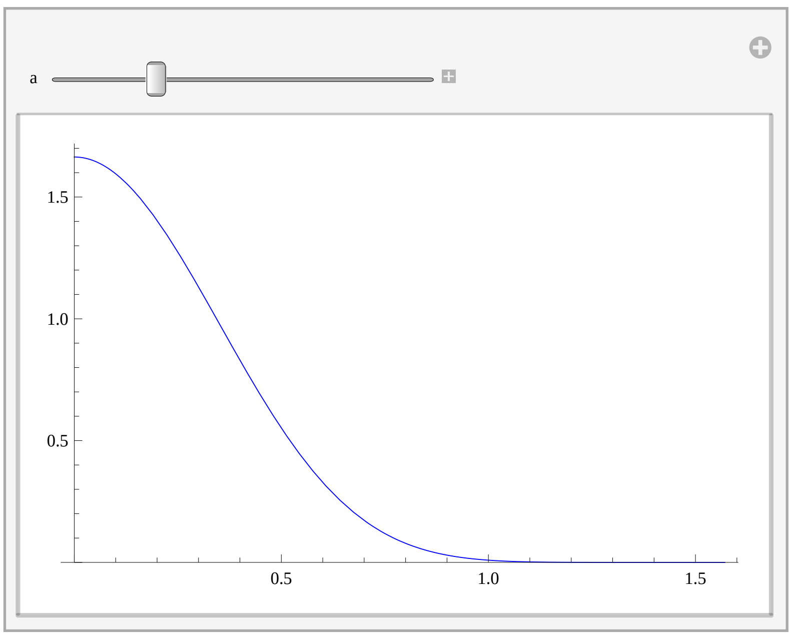
Out[13]=



And an interactive graph:

In[14]:=
```
Manipulate[
 Plot[phong /. αp → (8000^a - 1), {θ, 0, π / 2}, PlotStyle → pCol], {{a, 0.25}, 0, 1}]
```

Out[14]=



Beckmann NDF

This is the unnormalized Beckmann distribution function:

In[15]:= unnormalizedbeckmann = $$\frac{1}{\alpha b^2\,\mathrm{Cos}[\theta]^4}\, e^{-\left(\frac{1-\mathrm{Cos}[\theta]^2}{\mathrm{Cos}[\theta]^2\,\alpha b^2}\right)};$$

Compute the normalization factor (relative to projected area) for the Beckmann distribution function:

In[16]:=
```
beckmannnormf = Integrate[unnormalizedbeckmann Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αb > 0}]
```

Out[16]= π

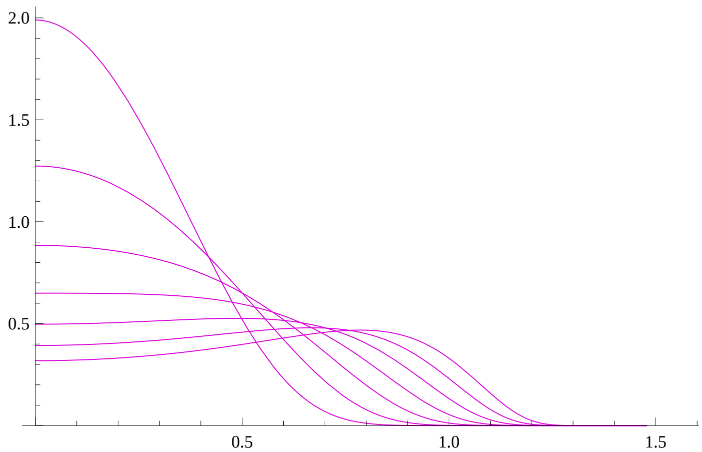
We see here that the correct normalization factor for the Beckmann distribution, given normalization over projected area, is $\frac{1}{\pi}$.

In[17]:=
```
beckmann = unnormalizedbeckmann / beckmannnormf
```

Out[17]=
$$\frac{e^{-\frac{\left(1-\mathrm{Cos}[\theta]^2\right)\mathrm{Sec}[\theta]^2}{\alpha b^2}}\,\mathrm{Sec}[\theta]^4}{\pi\,\alpha b^2}$$

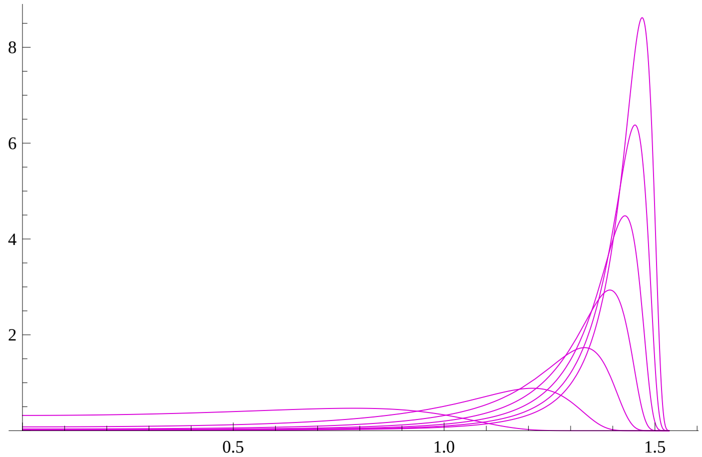
The Beckmann α_b parameter is equal to the RMS (root mean square) microfacet slope. Therefore its valid range is from 0 (non-inclusive – 0 corresponds to a perfect mirror or Dirac delta and causes divide by 0 errors in the Beckmann formulation) and up to arbitrarily high values. There is no special significance to a value of 1 – this just means that the RMS slope is 1/1 or 45°. We will look at the shape of the Beckmann NDF for moderately rough surfaces (*m* from 0.4 to 1):

In[18]:=
```
Plot[beckmann /. αb → # & /@ Range[0.4, 1.0, 0.1], {θ, 0, π / 2}, PlotStyle → bCol]
```

Out[18]=


We see here that at $\alpha_b$values above 0.75, a local minimum starts appearing at 0°. This is significantly different than a Phong or Gaussian lobe, where the "roughest" possible surface is a uniform distribution. The Beckmann distribution is qualitatively different in that its parameter is not related to the variance of the angle but the mean of the slope. Thus a "very rough" surface in the Beckmann context is not a uniform or almost-uniform distribution, but a distribution clustered around high slopes. Let us look at even larger values of *m*:

In[19]:=
```
Plot[beckmann /. αb → # & /@ Range[1, 7], {θ, 0, π / 2}, PlotStyle → bCol]
```

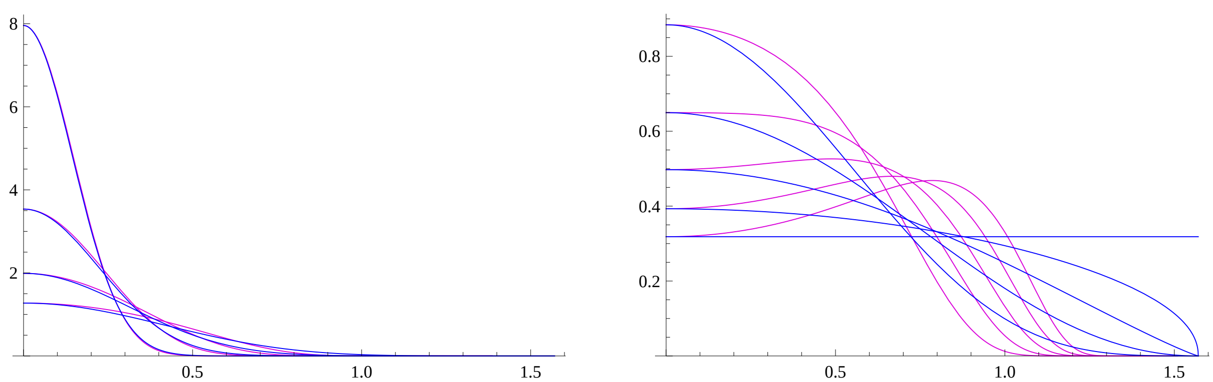
Out[19]=


This behavior is unfortunate for environment map prefiltering, since the frequency content of the NDF decreases to a certain roughness and then starts increasing with *m*. Beckmann is supposed to be a good match to real-world measurements, but I am not sure over what range of parameters these comparisons were carried out, and whether values this high (or even higher than 0.75, where the local minima starts appearing) are observed in practice.

Let us compare Beckmann and Phong, using an equivalence between the parameters of the two NDFs published in "Microfacet Models for Refraction through Rough Surfaces" (EGSR 2007) – note that the equivalence breaks down for $\alpha_b > 1$:

In[20]:=
$$\text{αb2αp}[\text{αb\_}] := \frac{2}{\text{αb}^2} - 2$$

In[21]:=
```
GraphicsRow[{Plot[{beckmann /. αb → # & /@ Range[0.2, 0.5, 0.1],
    phong /. αp → αb2αp[#] & /@ Range[0.2, 0.5, 0.1]},   {θ, 0, π / 2},
   PlotStyle → {bCol, pCol}], Plot[{beckmann /. αb → # & /@ Range[0.6, 1.0, 0.1],
    phong /. αp → αb2αp[#] & /@ Range[0.6, 1.0, 0.1]},
   {θ, 0, π / 2}, PlotStyle → {bCol, pCol}]}, ImageSize → Full]
```

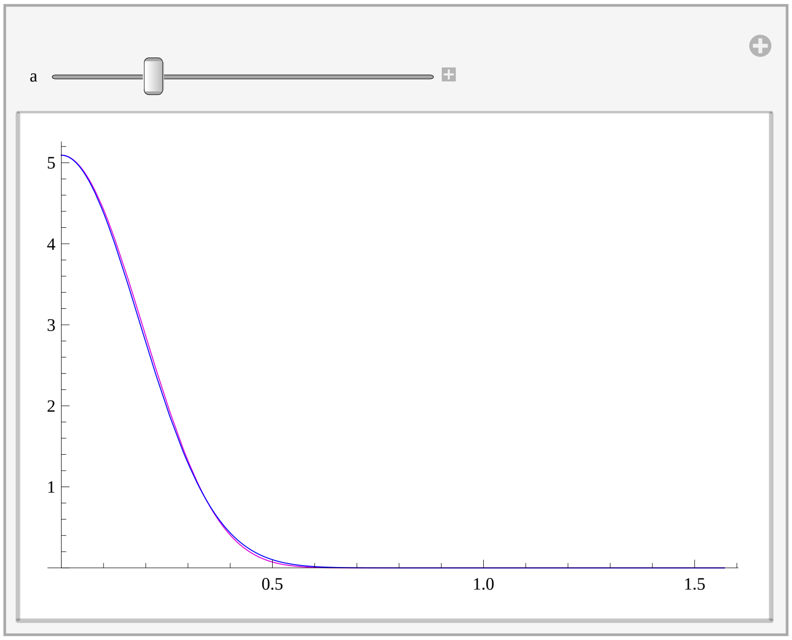
Out[21]=


For rough surfaces (left plot), the equivalence holds about as well as can be expected, but the shape of the NDFs starts to differ significantly as *m* increases. For relatively smooth surfaces (right plot) the two NDFs match surprisingly well. This is to Phong's credit, who devised his NDF (although not as such) purely from observation. As the value of α_b decreases, the match improves.

Interactive graph for “normal” (not super-rough) values, comparing with Phong:

In[22]:=
```
Manipulate[Plot[{beckmann /. αb → a, phong /. αp → αb2αp[a]},
  {θ, 0, π / 2}, PlotStyle → {bCol, pCol}], {{a, 0.25}, 0.01, 1.0}]
```

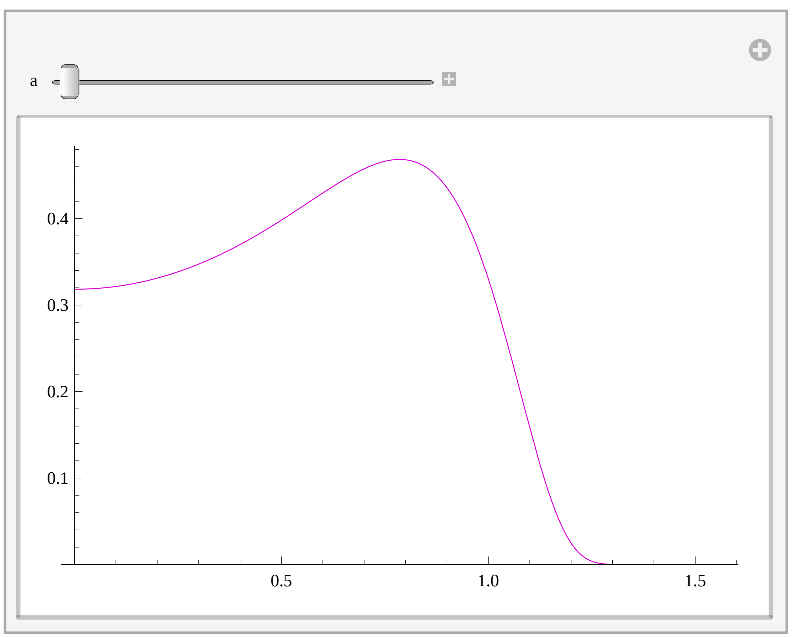
Out[22]=



Interactive graph for Beckmann by itself for super-rough values:

In[23]:=
```
Manipulate[Plot[beckmann /. αb → a, {θ, 0, π / 2}, PlotStyle → bCol], {a, 1.0, 10.0}]
```

Out[23]=



Torrance-Sparrow NDF

This NDF is a Gaussian on the angle between the microfacet normal and the macroscopic surface normal. We will need to normalize it since Torrance and Sparrow did not supply a normalization factor:

In[24]:= $\texttt{unnormalizedtorrancesparrow} = e^{-\left(\frac{\theta}{\alpha ts}\right)^2};$

We'll try for an analytical normalization factor:

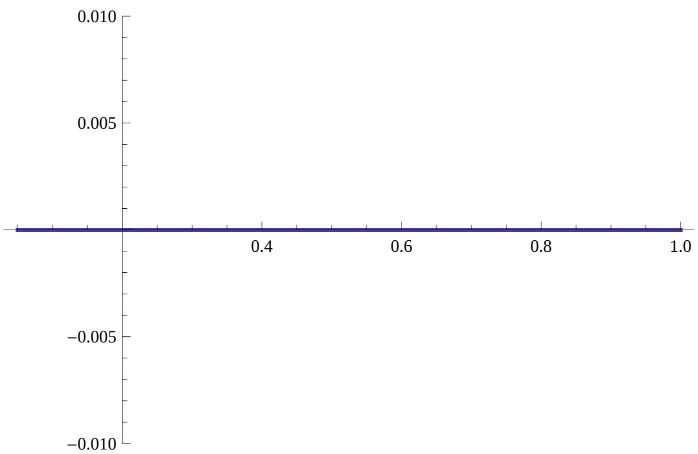
In[25]:=
```
normts = Integrate[unnormalizedtorrancesparrow Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αts > 0}]
```

Out[25]= $$\frac{1}{4} e^{-\alpha ts^2} \pi^{3/2} \alpha ts \left(-i \,\mathrm{Erf}\left[\frac{\pi}{2 \alpha ts} - i \alpha ts\right] + i \,\mathrm{Erf}\left[\frac{\pi}{2 \alpha ts} + i \alpha ts\right] + 2 \,\mathrm{Erfi}[\alpha ts]\right)$$

Wow, that's ugly! It also appears to be complex-valued, which is odd since the function being integrated was real-valued. Let's see if it is really complex-valued:

In[26]:=
```
Plot[Im[normts], {αts, 0.05, 1}, {PlotRange → {-0.01, 0.01}, PlotStyle → Thick}]
```

Out[26]=



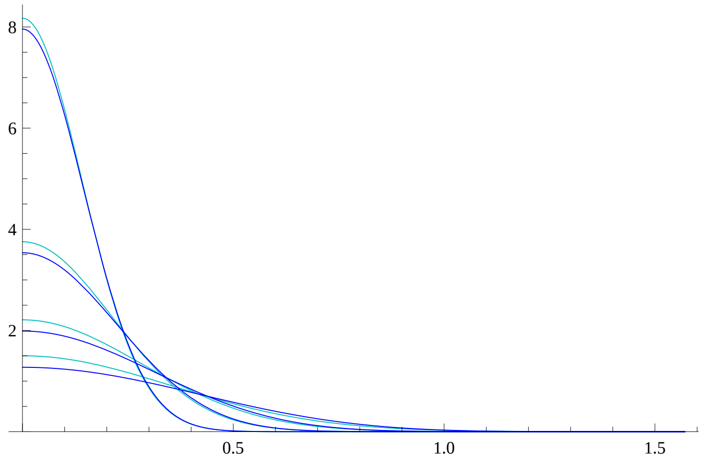
The imaginary part is 0 – looks like the normalization factor actually is real-valued and *Mathematica* is just being weird. If we were actually going to use this, we would fit a cheap function to the curve instead of using the analytical expression. But since we are just comparing it to other NDFs, no need to do that. Let's compare it to Blinn-Phong, using the Beckmann parameter conversion (according to the Cook-Torrance paper, the Beckmann and Torrance-Sparrow parameterizations are the same – both are equal to RMS slope):

In[27]:= $$\texttt{torrancesparrow} = \frac{\texttt{unnormalizedtorrancesparrow}}{\texttt{Re[normts]}}$$

Out[27]= $$\left(4 e^{-\frac{\theta^2}{\alpha ts^2}}\right) \Big/ \left(\pi^{3/2} \,\mathrm{Re}\left[e^{-\alpha ts^2} \alpha ts \left(-i \,\mathrm{Erf}\left[\frac{\pi}{2 \alpha ts} - i \alpha ts\right] + i \,\mathrm{Erf}\left[\frac{\pi}{2 \alpha ts} + i \alpha ts\right] + 2 \,\mathrm{Erfi}[\alpha ts]\right)\right]\right)$$

In[28]:=
```
Plot[{torrancesparrow /. αts → # & /@ Range[0.2, 0.5, 0.1],
  phong /. αp → αb2αp[#] & /@ Range[0.2, 0.5, 0.1]},
 {θ, 0, π / 2}, PlotStyle → {tsCol, pCol}]
```

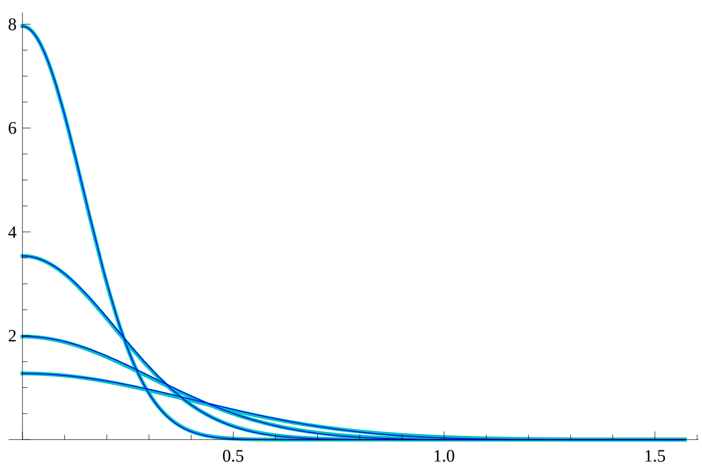
Out[28]=



The curves do look similar, but it appears that the equivalence between the parameterizations of the two distributions is a bit different than the one implied in the Cook-Torrance paper. We could work out the exact equivalence, but if the Torrance-Sparrow NDF turns out to have similar behavior to Phong over the whole range then it would be wasted effort since there would be no reason to use the (much more expensive) Torrance-Sparrow NDF. Let’s adjust parameter values manually to make the peaks coincide:

In[29]:=
```
Plot[{torrancesparrow /. αts → # & /@ {0.2027, 0.3097, 0.425, 0.552},
  phong /. αp → αb2αp[#] & /@ Range[0.2, 0.5, 0.1]},
 {θ, 0, π / 2}, PlotStyle → {{tsCol, Thick}, pCol}]
```

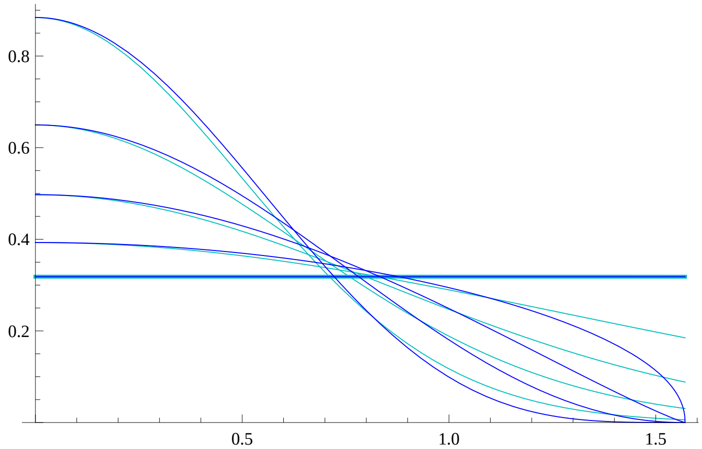
Out[29]=



The curves appear to be extremely close. Let’s look at a rougher part of the domain:

In[30]:=
```
Plot[{torrancesparrow /. αts → # & /@ {0.7035, 0.899, 1.195, 1.81},
  torrancesparrow /. αts → 100.0, phong /. αp → αb2αp[#] & /@ Range[0.6, 1.0, 0.1]},
 {θ, 0, π / 2}, PlotStyle → {tsCol, {tsCol, Thick}, pCol}]
```

Out[30]=



All in all, the behavior appears to be very similar to Phong. The curves for rough surfaces are a bit higher at glancing angles, but the overall trend is towards a uniform distribution, like Phong (and unlike Beckmann). Given this similarity in behavior and the much higher computational complexity of the Torrance-Sparrow NDF (even higher than it appears as first, since it uses the angle directly rather than the cosine), there does not appear to be a reason to use it.

Trowbridge-Reitz NDF

The original paper by Trowbridge and Reitz, the 1977 Blinn paper, and the 2007 paper by Walter et al. (where they refer to it as “the GGX distribution”) all have slightly different forms of this NDF. They are all equivalent other than constant factors; we will independently derive the normalization factor here:

In[31]:=
$$\text{unnormalizedtrowbridgereitz} = \frac{1}{\left(\text{Cos}[\theta]^2 \left(\alpha\text{tr}^2 - 1\right) + 1\right)^2};$$

In[32]:=
```
trowbridgereitznormf = Integrate[unnormalizedtrowbridgereitz Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αtr > 0}]
```

Out[32]=
$$\frac{\pi}{\alpha\text{tr}^2}$$

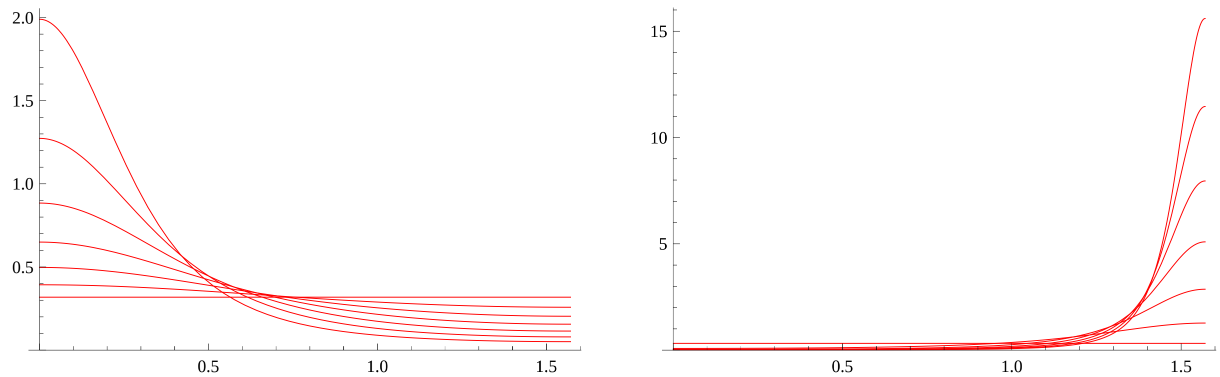
In[33]:=
$$\text{trowbridgereitz} = \frac{\text{unnormalizedtrowbridgereitz}}{\text{trowbridgereitznormf}}$$

Out[33]=
$$\frac{\alpha\text{tr}^2}{\pi \left(1 + \left(-1 + \alpha\text{tr}^2\right) \text{Cos}[\theta]^2\right)^2}$$

We’ll look at the distribution curves for moderate parameter values (on the left) as well as for high parameter values (on the right):

In[34]:=
```
GraphicsRow[{Plot[trowbridgereitz /. αtr → # & /@ Range[0.4, 1.0, 0.1], {θ, 0, π / 2},
    PlotStyle → trCol], Plot[trowbridgereitz /. αtr → # & /@ Range[1, 7],
    {θ, 0, π / 2}, PlotStyle → trCol]}, ImageSize → Full]
```

Out[34]=

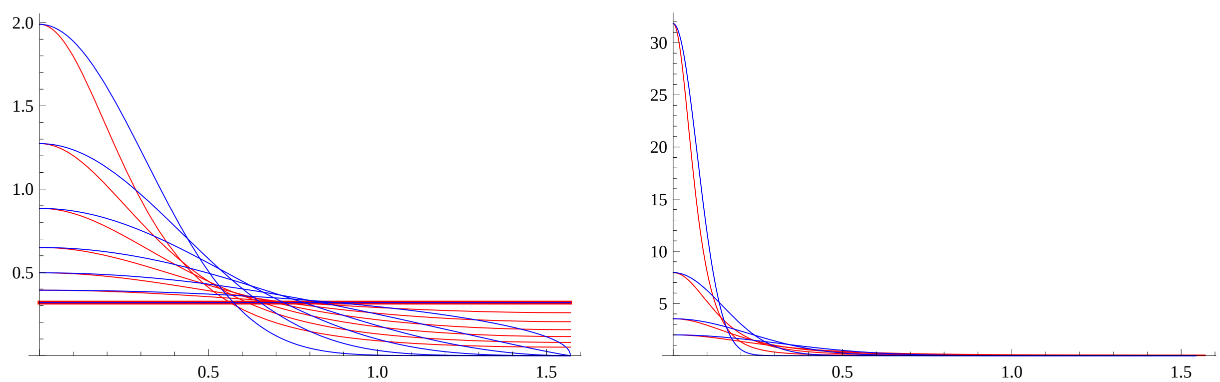


On the left, we see that the parameterization behaves approximately like Beckmann's: higher is rougher. Unlike Beckmann, a value of 1.0 gives a uniform distribution (flat line). On the right, we see that the Trowbridge-Reitz distribution also supports "super-rough" distributions, like Beckmann.

Let's compare Trowbridge-Reitz to Phong for the rough-to-moderate range (on the left) and for smoother surfaces (on the right). We use the Beckmann parameter equivalence, since behavior with respect to the parameterization appears similar:

In[35]:=
```
GraphicsRow[{Plot[{trowbridgereitz /. αtr → # & /@ Range[0.4, 0.9, 0.1],
     trowbridgereitz /. αtr → 1.0, phong /. αp → αb2αp[#] & /@ Range[0.4, 1.0, 0.1]},
    {θ, 0, π / 2}, PlotStyle → {trCol, {trCol, Thick}, pCol}],
  Plot[{trowbridgereitz /. αtr → # & /@ Range[0.1, 0.4, 0.1],
     phong /. αp → αb2αp[#] & /@ Range[0.1, 0.4, 0.1]},
    {θ, 0, π / 2}, PlotStyle → {trCol, pCol}]}, ImageSize → Full]
```

Out[35]=

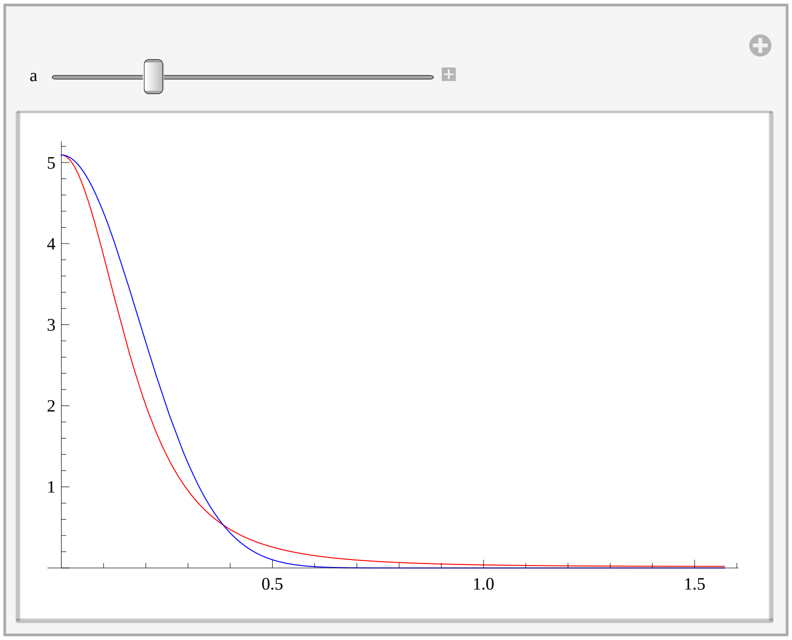


The distributions are somewhat similar, but the Trowbridge-Reitz distribution seems to have narrower peaks and longer "tails" across the entire range (except for the uniform distribution which is identical for both).

Finally, here's an interactive plot comparing it to Phong:

In[36]:=
```
Manipulate[Plot[{trowbridgereitz /. αtr → a, phong /. αp → αb2αp[a]},
  {θ, 0, π / 2}, PlotStyle → {trCol, pCol}], {{a, 0.25}, 0.01, 1.0}]
```

Out[36]=



ABC NDF

In[37]:=
```
unnormalizedabc = 1 / (1 + αabc (1 - Cos[θ]))^γabc;
```

In[38]:=
```
abcnormf = Integrate[unnormalizedabc Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αabc > 0, γabc > 0}]
```

Out[38]= $$\left(2\pi\,(1+\alpha abc)^{-\gamma abc}\left((1+\alpha abc)^2+(1+\alpha abc)^{\gamma abc}\,(-1+\alpha abc\,(-2+\gamma abc))\right)\right)\Big/\left(\alpha abc^2\,(-2+\gamma abc)\,(-1+\gamma abc)\right)$$

This function has singularities at $\gamma abc = 1.0$ and $\gamma abc = 2.0$. However, we can generate specific normalization terms for these values. These are the same as the limits of the more general function in the neighborhood of these values, indicating that the singularities are removable:

In[39]:=
```
abcnormfg1 = Integrate[(unnormalizedabc /. γabc → 1) Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αabc > 0}]
```

Out[39]= $$\frac{1}{\alpha abc^2}\,2\pi\,(-\alpha abc+(1+\alpha abc)\,\mathrm{Log}[1+\alpha abc])$$

In[40]:=
```
Limit[abcnormf, γabc → 1]
```

Out[40]= $$\frac{1}{\alpha abc^2}\,2\pi\,(-\alpha abc+(1+\alpha abc)\,\mathrm{Log}[1+\alpha abc])$$

In[41]:=
```
abcnormfg2 = Integrate[(unnormalizedabc /. γabc → 2) Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αabc > 0}]
```

Out[41]=
$$\frac{2\,\pi\,(\alpha abc - \mathrm{Log}[1 + \alpha abc])}{\alpha abc^2}$$

In[42]:=
```
Limit[abcnormf, γabc → 2]
```

Out[42]=
$$\frac{2\,\pi\,(\alpha abc - \mathrm{Log}[1 + \alpha abc])}{\alpha abc^2}$$

We'll create a piecewise form of the function covering the singularity cases, so that we can easily plot the normalized NDF over a range of parameter values:

In[43]:=
```
abcnormfpw = Piecewise[{{abcnormfg1, γabc == 1}, {abcnormfg2, γabc == 2}}, abcnormf]
```

Out[43]=
$$\begin{cases} \frac{2\,\pi\,(-\alpha abc + (1+\alpha abc)\,\mathrm{Log}[1+\alpha abc])}{\alpha abc^2} & \gamma abc == 1 \\ \frac{2\,\pi\,(\alpha abc - \mathrm{Log}[1+\alpha abc])}{\alpha abc^2} & \gamma abc == 2 \\ \left(2\,\pi\,(1+\alpha abc)^{-\gamma abc}\left((1+\alpha abc)^2 + (1+\alpha abc)^{\gamma abc}\,(-1+\alpha abc\,(-2+\gamma abc))\right)\right) \Big/ \left(\alpha abc^2\,(-2+\gamma abc)\,(-1+\gamma abc)\right) & \mathrm{True} \end{cases}$$

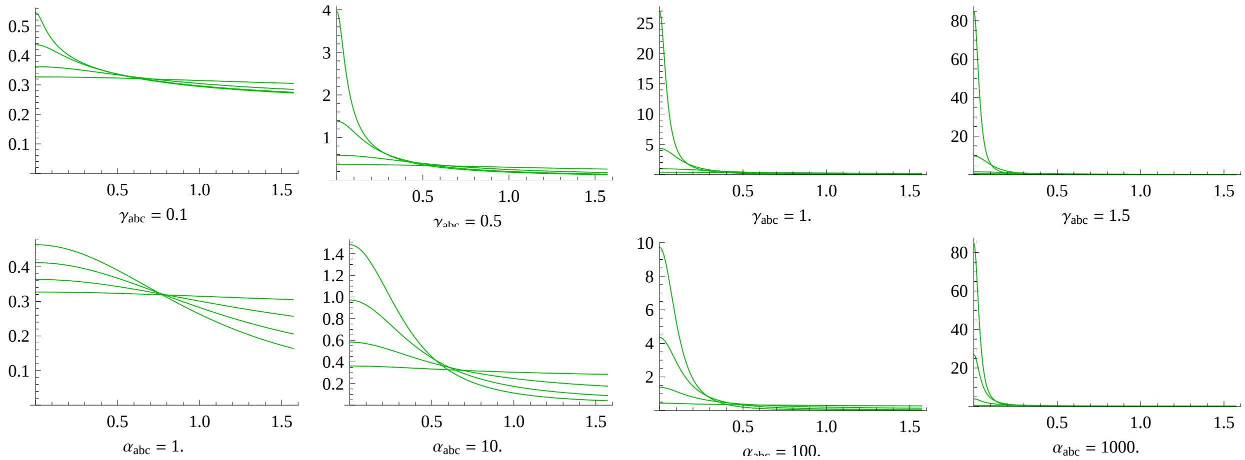
In[44]:=
$$abc = \frac{unnormalizedabc}{abcnormfpw}$$

Out[44]=
$$(1 + \alpha abc\,(1 - \mathrm{Cos}[\theta]))^{-\gamma abc} \Big/ \begin{cases} \frac{1}{\alpha abc^2}\,2\,\pi\,(-\alpha abc + (1+\alpha abc)\,\mathrm{Log}[1+\alpha abc]) & \gamma abc == 1 \\ \frac{2\,\pi\,(\alpha abc - \mathrm{Log}[1+\alpha abc])}{\alpha abc^2} & \gamma abc == 2 \\ \left(2\,\pi\,(1+\alpha abc)^{-\gamma abc}\left((1+\alpha abc)^2 + (1+\alpha abc)^{\gamma abc}\,(-1+\alpha abc\,(-2+\gamma abc))\right)\right) \Big/ \left(\alpha abc^2\,(-2+\gamma abc)\,(-1+\gamma abc)\right) & \mathrm{True} \end{cases}$$

This normalization term has an extremely complicated form - definitely too expensive for games; even for film use, it would be convenient to simplify it. Since when plotted it yields smooth curves, it should be possible to create a much simpler approximation for production use. Let us look at the behavior of the normalized ABC NDF as its parameters are varied. Since the parameter space is two-dimensional, we'll need more plots than in the previous sections:

In[45]:=
```
plot1abc[x_] := Labeled[Plot[abc /. {αabc → #, γabc → x} & /@ {1.0, 10.0, 100.0, 1000.0},
    {θ, 0, π / 2}, PlotStyle → abcCol], "γabc  = " <> ToString[x]]
plot2abc[x_] := Labeled[Plot[abc /. {αabc → x, γabc → #} & /@ {0.1, 0.5, 1.0, 1.5},
    {θ, 0, π / 2}, PlotStyle → abcCol], "αabc  = " <> ToString[x]]
GraphicsGrid[{plot1abc /@ {0.1, 0.5, 1.0, 1.5},
  plot2abc /@ {1.0, 10.0, 100.0, 1000.0}}, ImageSize → Full, AspectRatio → 0.25]
```

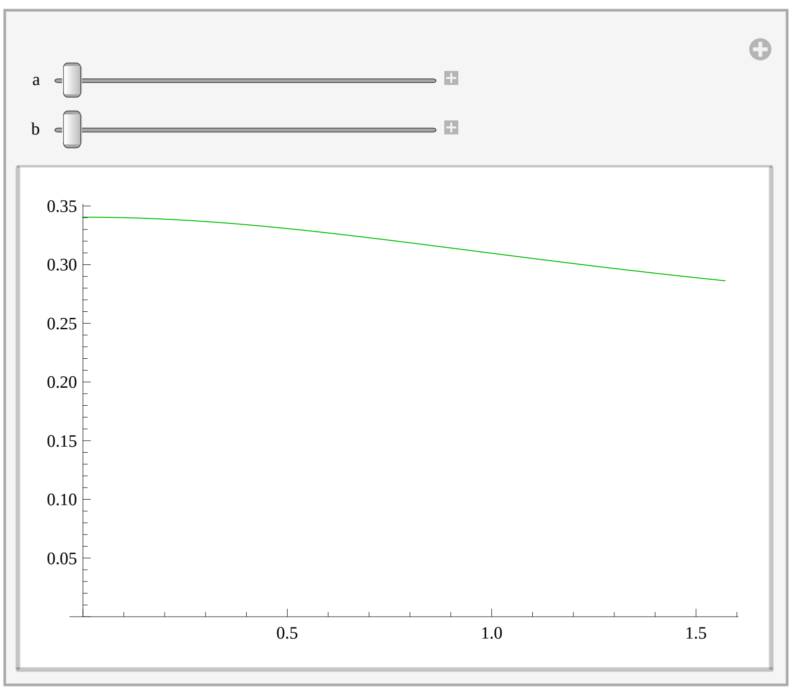
Out[47]=



And here’s an interactive plot:

In[48]:=
```
Manipulate[Plot[abc /. {αabc → a, γabc → b}, {θ, 0, π / 2}, PlotStyle → abcCol],
 {a, 1.0, 1000.0}, {b, 0.25, 2.5}]
```

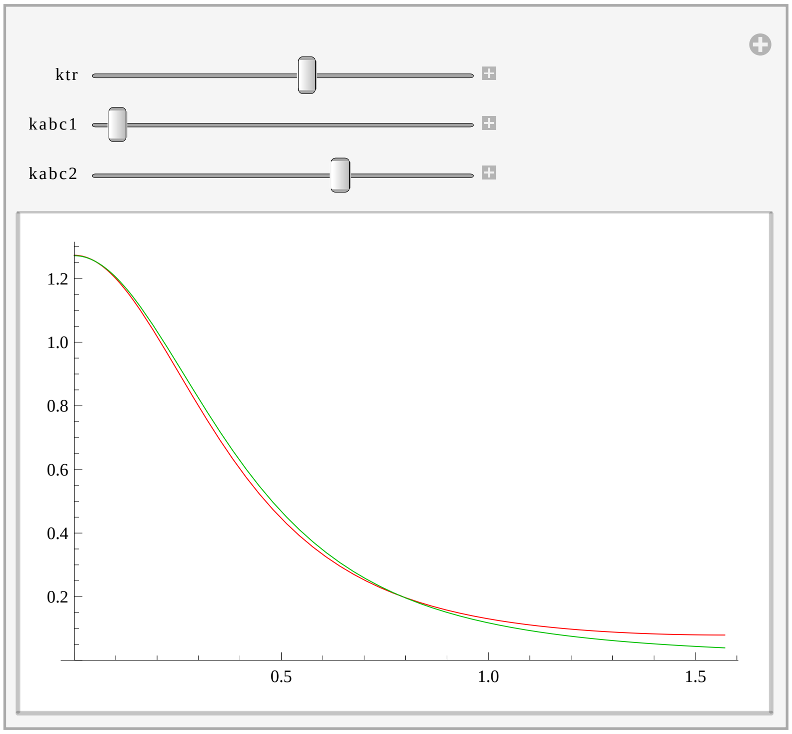
Out[48]=



Experimenting with various values shows us that the value of γabc appears to control the shape, while the value of αabc controls the roughness. (They are not cleanly separated, so when varying γabc you need to change αabc to keep the same roughness.) Let’s see if we can fit Trowbridge-Reitz using ABC:

In[49]:=
```
Manipulate[Plot[{trowbridgereitz /. αtr → ktr, abc /. {αabc → kabc1, γabc → kabc2}},
  {θ, 0, π / 2}, PlotStyle → {trCol, abcCol}], {{ktr, 0.5}, 0.1, 0.8},
 {{kabc1, 6.3}, 1.04, 254.0}, {{kabc2, 1.75}, 0.25, 2.5}]
```

Out[49]=

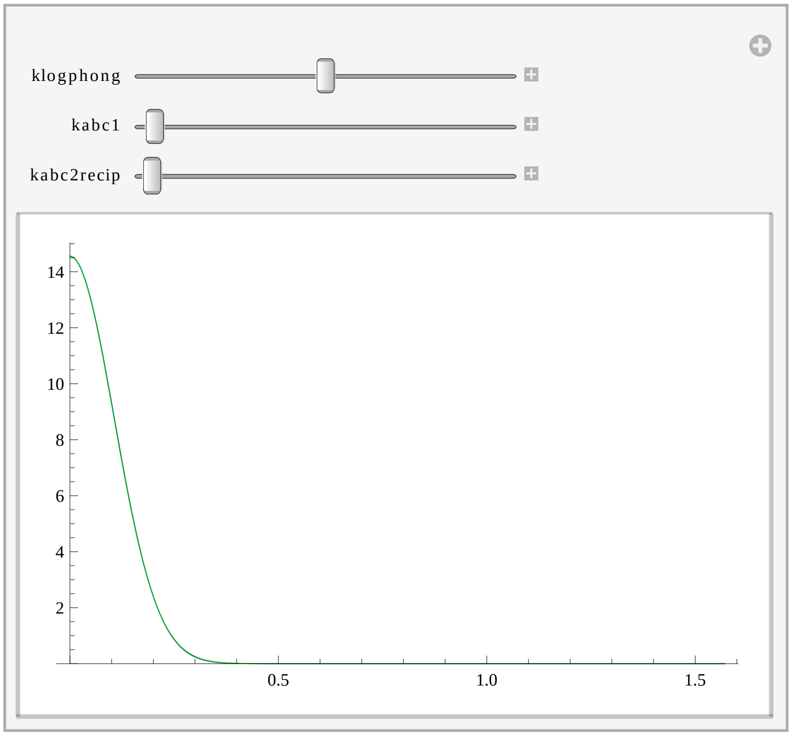


We can see that an γabc value of about 1.75 fits pretty well to Trowbridge-Reitz across the roughness range (less well for rough surfaces, better for smooth ones). Note that we don’t have an equivalence between them, so we just manually adjust the αabc parameter of the ABC curves until the peaks coincide with the Trowbridge-Reitz ones.

Now let’s try to fit Phong with ABC:

In[50]:=
```
Manipulate[
 Plot[{phong /. αp → 8000.0^klogphong, abc /. {αabc → kabc1, γabc → 1 / kabc2recip}},
  {θ, 0, π / 2}, PlotStyle → {pCol, abcCol}], {{klogphong, 0.5}, 0.0, 1.0},
 {{kabc1, 0.0905}, 0.0001, 10.0}, {{kabc2recip, 0.001}, 0.0001, 0.999}]
```

Out[50]=

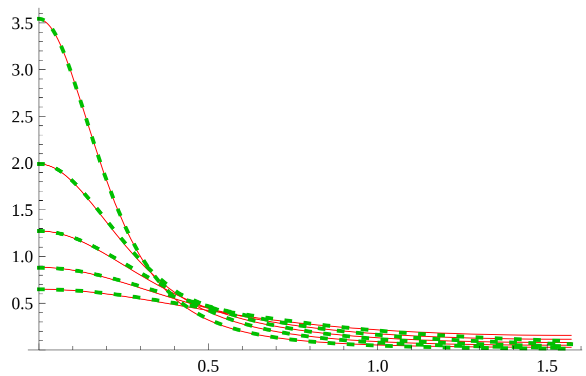


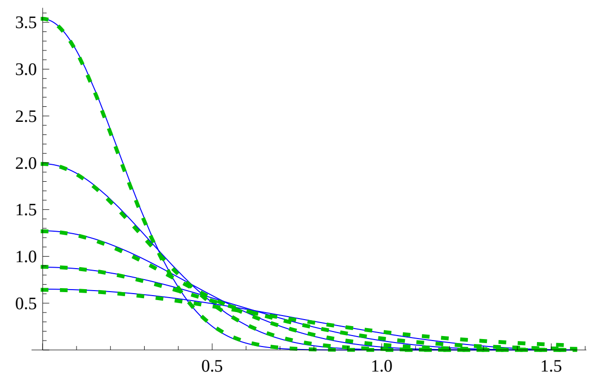
It seems that ABC asymptotically approaches Phong as the value of γabc approaches infinity (here we also lacked an equivalence so we adjusted αabc values manually until the peaks matched).

Let’s demonstrate ABC’s fit to Trowbridge-Reitz with a static plot for γabc = 1.75, and to Phong with a static plot for γabc set to a high value (1000):

In[51]:=
```
GraphicsRow[{Plot[{trowbridgereitz /. αtr → # & /@ Range[0.3, 0.7, 0.1],
     abc /. {αabc → #, γabc → 1.75} & /@ {23.5, 11.6, 6.3, 3.55, 2.0}}, {θ, 0, π / 2},
    PlotStyle → {trCol, {abcCol, Thick, AbsoluteDashing[{2, 8}]}}],
  Plot[{phong /. αp → αb2αp[#] & /@ Range[0.3, 0.7, 0.1],
     abc /. {αabc → #, γabc → 1000.0} & /@
      {0.0212, 0.0114, 0.0068, 0.0043, 0.0026}}, {θ, 0, π / 2},
    PlotStyle → {pCol, {abcCol, Thick, AbsoluteDashing[{2, 8}]}}]}, ImageSize → Full]
```

Out[51]=





Since they are not directly apparent from the Plot command, let’s see the range of Phong parameters covered in the right plot:

In[52]:=
```
αb2αp[0.3]
```

Out[52]= 20.2222

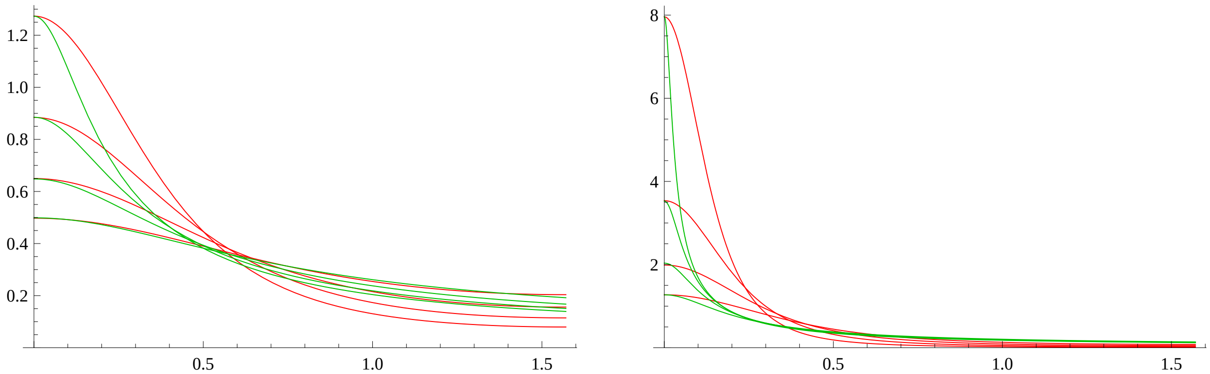
In[53]:=
```
αb2αp[0.7]
```

Out[53]= 2.08163

As we have seen, with an γabc value of 1.75, ABC can mimic Trowbridge-Reitz quite well. With higher values, ABC can approach the appearance of Phong. (It should be noted that these are much higher than any of the values fitted to the Matusik dataset by Low et al.; this may indicate that real-world materials do not typically exhibit Gaussian normal distributions.) With γabc values lower than 1.75, the ABC distribution is even “spikier” than Trowbridge-Reitz; we will look at a value of 0.5 (a relatively low value for the Matusik dataset fitting performed in the paper by Low et al. – lower values were only used for very rough surfaces), comparing it to Trowbridge-Reitz (manually adjusted so the peaks match):

In[54]:=
```
GraphicsRow[{Plot[{trowbridgereitz /. αtr → # & /@ Range[0.5, 0.8, 0.1],
    abc /. {αabc → #, γabc → 0.5} & /@ {82.5, 33.5, 14.0, 5.75}},
   {θ, 0, π / 2}, PlotStyle → {trCol, abcCol}],
  Plot[{trowbridgereitz /. αtr → # & /@ Range[0.2, 0.5, 0.1],
    abc /. {αabc → #, γabc → 0.5} & /@ {4250, 785, 240, 82.5}},
   {θ, 0, π / 2}, PlotStyle → {trCol, abcCol}]}, ImageSize → Full]
```

Out[54]=



We see that with an γabc value of 0.5, ABC is significantly “spikier” than Trowbridge-Reitz when modeling rough surfaces (on the left), and extremely so when modeling smooth ones (on the right).

Shifted Gamma Distribution

In[55]:=
$$\text{p22}[\text{x\_}] := \frac{\alpha\text{sgd}^{\gamma\text{sgd}-1}}{\text{Gamma}[1-\gamma\text{sgd},\ \alpha\text{sgd}]}\ \frac{e^{-\frac{\alpha\text{sgd}^2+\text{x}}{\alpha\text{sgd}}}}{\left(\alpha\text{sgd}^2+\text{x}\right)^{\gamma\text{sgd}}}$$

In[56]:=
$$\text{sgd} = \frac{\text{p22}\left[\frac{1-\text{Cos}[\theta]^2}{\text{Cos}[\theta]^2}\right]}{\pi\ \text{Cos}[\theta]^4}$$

Out[56]=
$$\left(e^{-\frac{\alpha\text{sgd}^2+\left(1-\text{Cos}[\theta]^2\right)\text{Sec}[\theta]^2}{\alpha\text{sgd}}}\ \alpha\text{sgd}^{-1+\gamma\text{sgd}}\ \text{Sec}[\theta]^4\ \left(\alpha\text{sgd}^2+\left(1-\text{Cos}[\theta]^2\right)\text{Sec}[\theta]^2\right)^{-\gamma\text{sgd}}\right) \Big/ \left(\pi\ \text{Gamma}[1-\gamma\text{sgd},\ \alpha\text{sgd}]\right)$$

First, let’s confirm that it’s normalized, using an analytical integral:

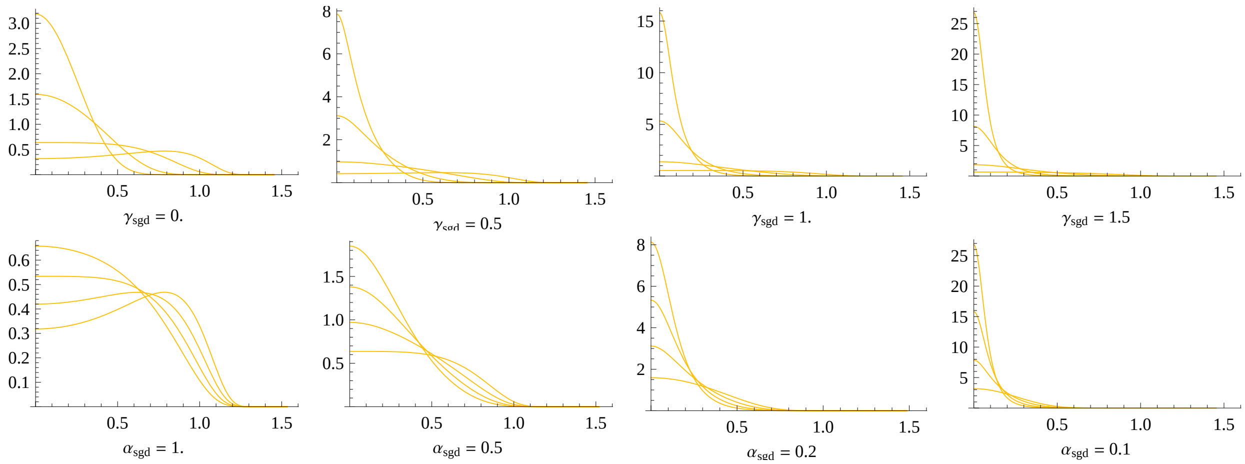
In[57]:=
```
sgdnormf = Integrate[sgd Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αsgd > 0, γsgd > 0}]
```

Out[57]= 1

Yes, it’s normalized. Let’s take a look at various parameter values, spanning the rough-to-moderate part of the range of values used for fitting SGD to the Matusik database:

In[58]:=
```
plot1sgd[x_] := Labeled[Plot[sgd /. {αsgd → #, γsgd → x} & /@ {1.0, 0.5, 0.2, 0.1},
    {θ, 0, π / 2}, PlotStyle → sgdCol], "γsgd = " <> ToString[x]]
plot2sgd[x_] := Labeled[Plot[sgd /. {αsgd → x, γsgd → #} & /@ {0.0, 0.5, 1.0, 1.5},
    {θ, 0, π / 2}, PlotStyle → sgdCol], "αsgd = " <> ToString[x]]
GraphicsGrid[{plot1sgd /@ {0.0, 0.5, 1.0, 1.5}, plot2sgd /@ {1.0, 0.5, 0.2, 0.1}},
 ImageSize → Full, AspectRatio → 0.25]
```

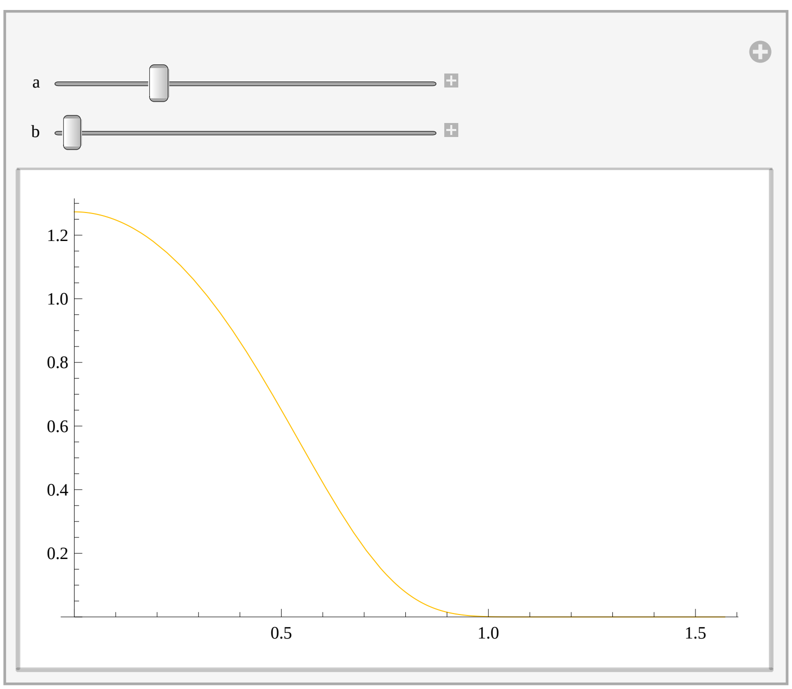
Out[60]=



Let's look at an interactive graph with the parameters covering the range of fitted values for the Matusik database:

In[61]:=
```
Manipulate[Plot[sgd /. {αsgd → a, γsgd → b}, {θ, 0, π / 2}, PlotStyle → sgdCol],
 {{a, 0.25}, 0.0001, 1.0}, {b, 0.0, 1.5}]
```

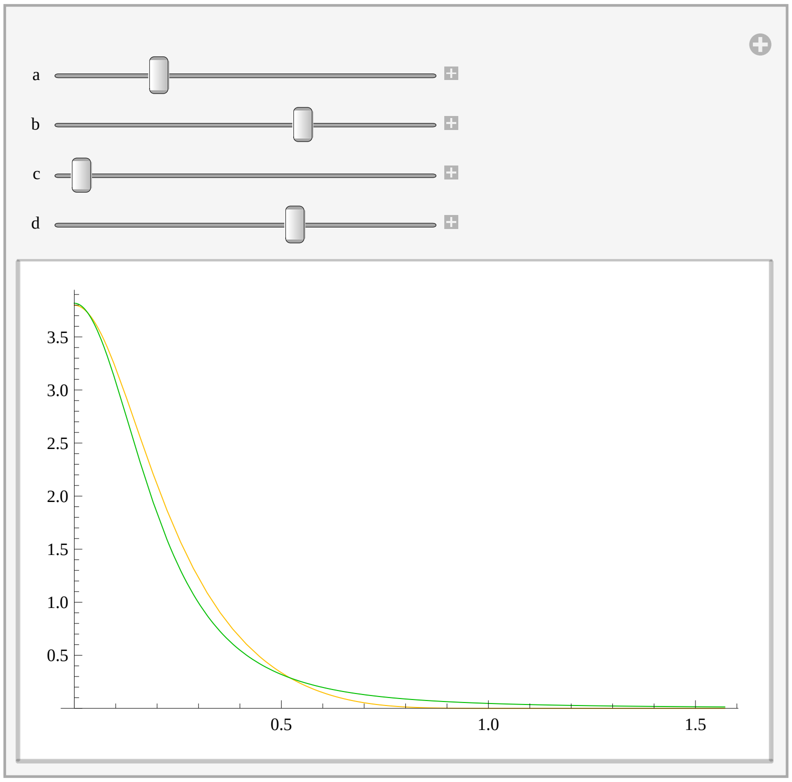
Out[61]=



Finally, let's compare it with ABC:

In[62]:=
```
Manipulate[Plot[{sgd /. {αsgd → a, γsgd → b}, abc /. {αabc → c, γabc → d}},
  {θ, 0, π / 2}, PlotStyle → {sgdCol, abcCol}], {{a, 0.25}, 0.0001, 1.0},
 {{b, 1.0}, 0.0, 1.5}, {{c, 27.0}, 1.0, 1000.0}, {{d, 1.7}, 0.25, 2.5}]
```

Out[62]=



We see that the SGD NDF goes quickly to 0.0 even for moderately smooth surfaces and cannot replicate the “long tails” that ABC can produce. We also see that unlike ABC, the SGD NDF can model slightly “super-rough” surfaces, though it’s unclear how useful this feature is (also, the combination of parameters that produces this behavior is not found in the material fitting performed by Bagher et al.).

Generalized Trowbridge-Reitz NDF

In[63]:=

$$\text{unnormalizedgtr} = \frac{1}{\left(\text{Cos}[\theta]^2 \left(\alpha\text{gtr}^2 - 1\right) + 1\right)^{\gamma\text{gtr}}};$$

In[64]:=
```
gtrnormf = Integrate[unnormalizedgtr Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αgtr > 0, γgtr > 0}]
```

Out[64]=

$$-\frac{\pi \left(-1 + \alpha\text{gtr}^{2-2\gamma\text{gtr}}\right)}{\left(-1 + \alpha\text{gtr}^2\right) \left(-1 + \gamma\text{gtr}\right)}$$

This function has singularities at γgtr = 1.0 and αgtr = 1.0. However, we can generate specific normalization terms for these values. These are the same as the limits of the more general function in the neighborhood of these values, indicating that the singularities are removable:

In[65]:=
```
gtrnormfg1 = Integrate[(unnormalizedgtr /. γgtr → 1) Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {αgtr > 0}]
```

Out[65]= $\frac{2\pi\,\mathrm{Log}[\alpha\mathrm{gtr}]}{-1+\alpha\mathrm{gtr}^2}$

In[66]:=
```
Limit[gtrnormf, γgtr → 1]
```

Out[66]= $\frac{2\pi\,\mathrm{Log}[\alpha\mathrm{gtr}]}{-1+\alpha\mathrm{gtr}^2}$

In[67]:=
```
gtrnormfa1 = Integrate[(unnormalizedgtr /. αgtr → 1) Sin[θ] Cos[θ],
  {ϕ, -π, π}, {θ, 0, π / 2}, Assumptions → {γgtr > 0}]
```

Out[67]= π

In[68]:=
```
Limit[gtrnormf, αgtr → 1]
```

Out[68]= π

We'll create a piecewise form of the function covering the singularity cases, so that we can easily plot the normalized NDF over a range of parameter values:

In[69]:=
```
gtrnormfpw = Piecewise[{{gtrnormfa1, αgtr == 1}, {gtrnormfg1, γgtr == 1}}, gtrnormf]
```

Out[69]= $$\begin{cases} \pi & \alpha\mathrm{gtr} == 1 \\ \frac{2\pi\,\mathrm{Log}[\alpha\mathrm{gtr}]}{-1+\alpha\mathrm{gtr}^2} & \gamma\mathrm{gtr} == 1 \\ -\frac{\pi\left(-1+\alpha\mathrm{gtr}^{2-2\gamma\mathrm{gtr}}\right)}{\left(-1+\alpha\mathrm{gtr}^2\right)(-1+\gamma\mathrm{gtr})} & \mathrm{True} \end{cases}$$

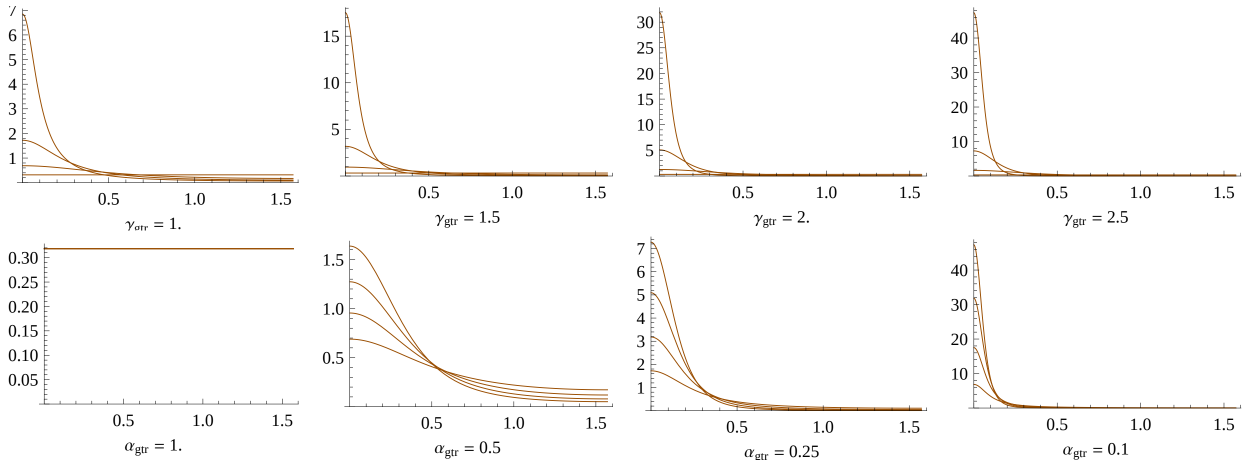
In[70]:=
$$\mathbf{gtr} = \frac{\mathbf{unnormalizedgtr}}{\mathbf{gtrnormfpw}}$$

Out[70]= $$\frac{\left(1+\left(-1+\alpha\mathrm{gtr}^2\right)\mathrm{Cos}[\theta]^2\right)^{-\gamma\mathrm{gtr}}}{\begin{cases} \pi & \alpha\mathrm{gtr} == 1 \\ \frac{2\pi\,\mathrm{Log}[\alpha\mathrm{gtr}]}{-1+\alpha\mathrm{gtr}^2} & \gamma\mathrm{gtr} == 1 \\ -\frac{\pi\left(-1+\alpha\mathrm{gtr}^{2-2\gamma\mathrm{gtr}}\right)}{\left(-1+\alpha\mathrm{gtr}^2\right)(-1+\gamma\mathrm{gtr})} & \mathrm{True} \end{cases}}$$

This normalization term has a somewhat complicated form, which is likely too expensive for performance-critical applications such as games. However, when plotted it yields smooth curves, indicating that it should be possible to create a simpler approximation for game use. Let us look at the behavior of the normalized GTR NDF as its parameters are varied.

In[71]:=
```
plot1gtr[x_] := Labeled[Plot[gtr /. {αgtr → #, γgtr → x} & /@ {1.0, 0.5, 0.25, 0.1},
    {θ, 0, π / 2}, PlotStyle → gtrCol], "γgtr = " <> ToString[x]]
plot2gtr[x_] := Labeled[Plot[gtr /. {αgtr → x, γgtr → #} & /@ {1.0, 1.5, 2.0, 2.5},
    {θ, 0, π / 2}, PlotStyle → gtrCol], "αgtr = " <> ToString[x]]
GraphicsGrid[{plot1gtr /@ {1.0, 1.5, 2.0, 2.5}, plot2gtr /@ {1.0, 0.5, 0.25, 0.1}},
 ImageSize → Full, AspectRatio → 0.25]
```

Out[73]=

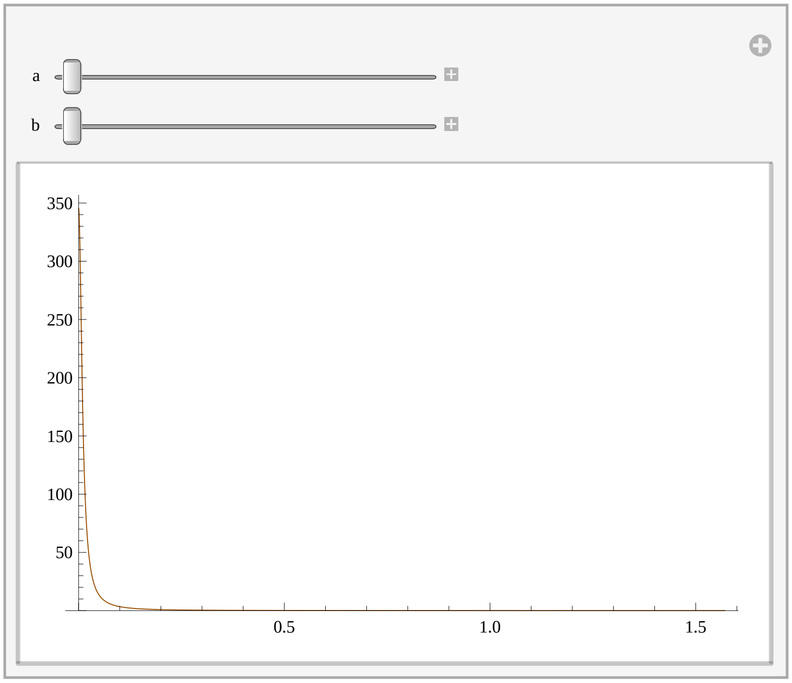


As a generalized form of Trowbridge-Reitz, GTR inherits some of its properties. When the αgtr parameter is equal to 1.0, we get the uniform (constant) distribution. Decreasing αgtr makes the surface smoother (creating narrower and more intense highlights. Increasing αgtr beyond 1.0 will create “super-rough” surfaces (this behavior is very similar to Trowbridge-Reitz, so we won’t bother plotting it). As with ABC, the separation of roughness and shape is not perfect, so when varying γgtr you need to change αgtr to keep the same roughness.)

And here’s an interactive plot:

In[74]:=
```
Manipulate[Plot[gtr /. {αgtr → a, γgtr → b}, {θ, 0, π / 2}, PlotStyle → gtrCol],
 {a, 0.01, 1.0}, {b, 1.0, 2.5}]
```

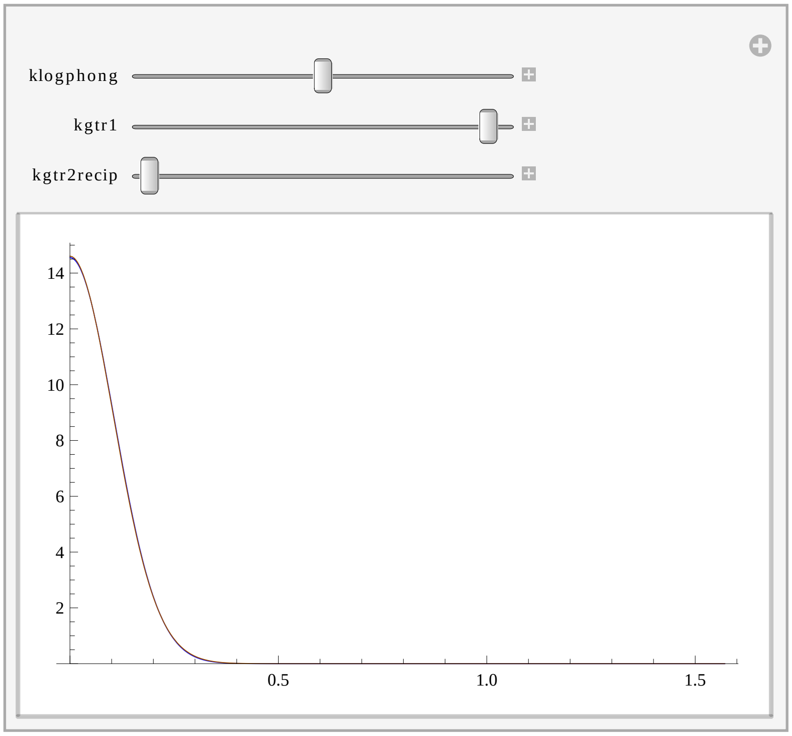
Out[74]=



Let’s see if we can fit Phong using GTR:

In[75]:=
```
Manipulate[
 Plot[{phong /. αp → 8000.0^klogphong, gtr /. {αgtr → kgtr1, γgtr → 1 / kgtr2recip}},
  {θ, 0, π / 2}, PlotStyle → {pCol, gtrCol}], {{klogphong, 0.5}, 0.0, 1.0},
 {{kgtr1, 0.9778}, 0.01, 1.0}, {{kgtr2recip, 0.001}, 0.0001, 0.999}]
```

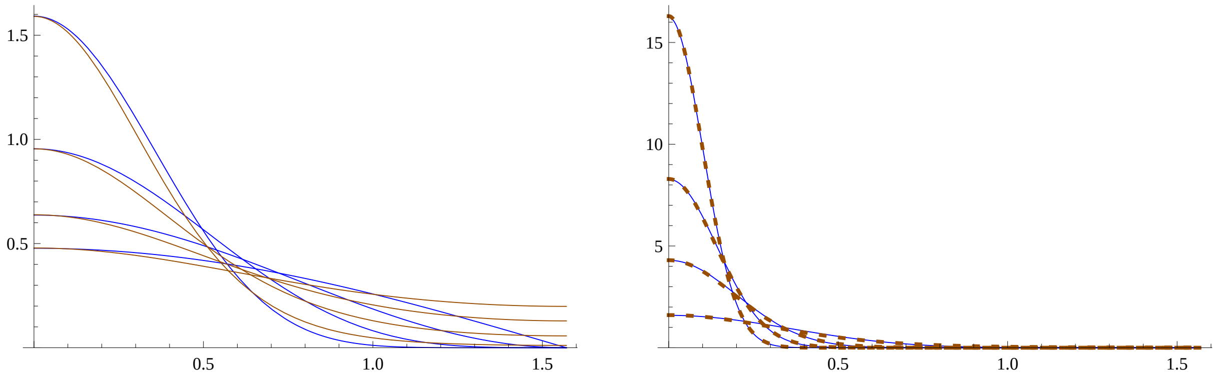
Out[75]=



Similarly to ABC, it appears that GTR asymptotically approaches Phong as the value of γgtr approaches infinity (and also as with ABC, we don't have a known parameter equivalence so we adjusted αgtr values manually until the peaks matched). Let's demonstrate the match with a very high γgtr value (10000) on a pair of static plots (one covering smoother surfaces and one covering rougher ones):

In[76]:=
```
GraphicsRow[{Plot[{phong /. αp → # & /@ {8, 4, 2, 1},
    gtr /. {αgtr → #, γgtr → 10 000} & /@ {0.999752, 0.999859, 0.99992, 0.999956}},
   {θ, 0, π / 2}, PlotStyle → {pCol, gtrCol}],
  Plot[{phong /. αp → # & /@ {100, 50, 25, 8}, gtr /. {αgtr → #, γgtr → 10 000} & /@
     {0.99745, 0.9987, 0.999325, 0.99975}}, {θ, 0, π / 2},
   PlotStyle → {pCol, {gtrCol, Thick, AbsoluteDashing[{2, 8}]}}]}, ImageSize → Full]
```

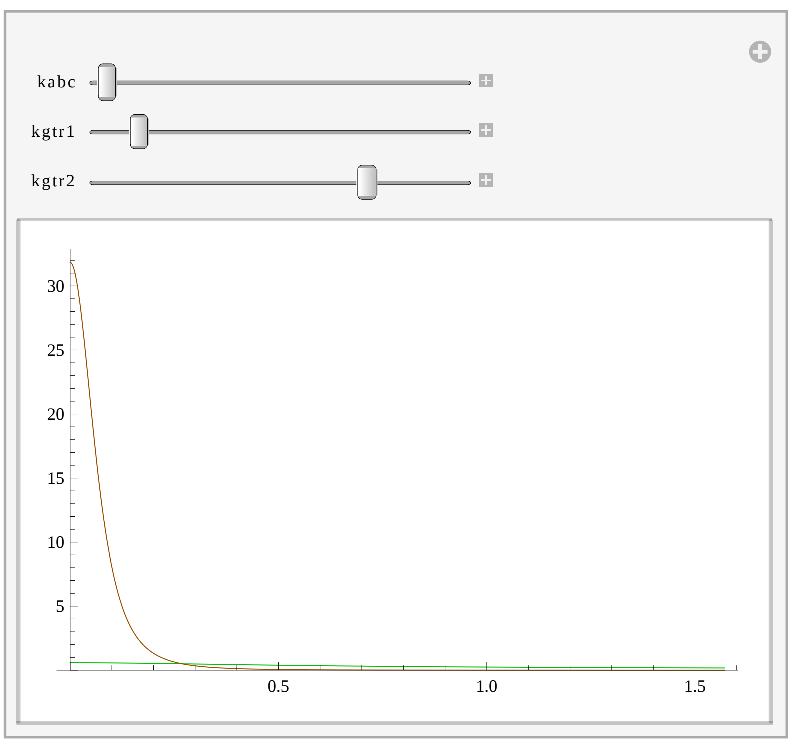
Out[76]=



Of course, GTR can match Trowbridge-Reitz exactly since setting γgtr to 2.0 makes the two exactly equivalent. Let's see if GTR can match the even "spikier" curves we get with ABC by setting γabc to 0.5:

In[77]:=
```
Manipulate[Plot[{abc /. {αabc → kabc, γabc → 0.5}, gtr /. {αgtr → kgtr1, γgtr → kgtr2}},
  {θ, 0, π / 2}, PlotStyle → {abcCol, gtrCol}], {{kabc, 10.0}, 5.0, 4000.0},
 {{kgtr1, 0.1}, 0.01, 1.0}, {{kgtr2, 2.0}, 0.5, 2.5}]
```

Out[77]=



We can see that an γgtr value of about 0.54 fits pretty well to ABC with γabc=0.5 across the roughness range (better for smoother surfaces, slightly less well for rough ones). Note that we don't have a known equivalence between them, so we just manually adjust the αgtr parameter of the GTR curves until the peaks coincide with the ABC ones. Let’s show the match with two static plots (one covering smoother surfaces and one covering rougher ones):

In[78]:=
```
GraphicsRow[{Plot[{abc /. {αabc → #, γabc → 0.5} & /@ {100, 30, 15, 5},
     gtr /. {αgtr → #, γgtr → 0.54} & /@ {0.146, 0.251, 0.338, 0.517}}, {θ, 0, π / 2},
    PlotStyle → {abcCol, {gtrCol, Thick, AbsoluteDashing[{2, 8}]}}],
   Plot[{abc /. {αabc → #, γabc → 0.5} & /@ {4000, 750, 250, 100},
     gtr /. {αgtr → #, γgtr → 0.54} & /@ {0.02625, 0.05725, 0.0955, 0.146}},
    {θ, 0, π / 2}, PlotStyle → {abcCol, {gtrCol, Thick, AbsoluteDashing[{2, 8}]}}]},
  ImageSize → Full]
```

Out[78]=