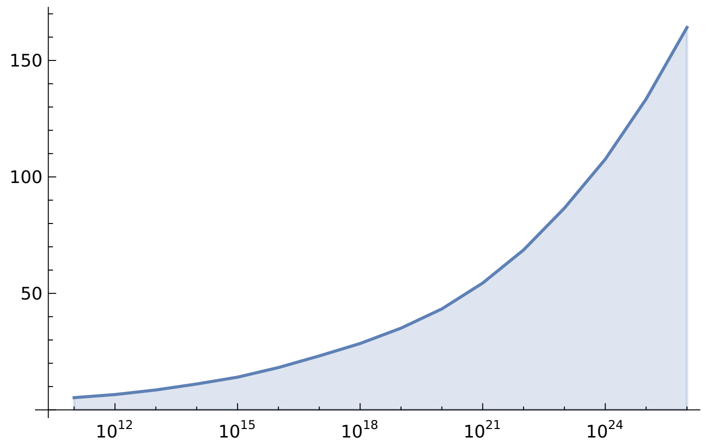
```
(* List of fast Gourdon alpha factors (alpha = alpha_y * alpha_z) found by
 running pi(x) benchmarks using the find_optimal_alpha_gourdon.sh script *)

In[1]:= alphaGourdon = {{10^11, 5.236}, {10^12, 6.571}, {10^13, 8.534}, {10^14, 11.096},
   {10^15, 14.031}, {10^16, 18.159}, {10^17, 23.143}, {10^18, 28.479},
   {10^19, 35.073}, {10^20, 43.327}, {10^21, 54.440}, {10^22, 68.642},
   {10^23, 86.600}, {10^24, 107.593}, {10^25, 133.439}, {10^26, 164.157}}

Out[1]= {{100 000 000 000, 5.236}, {1 000 000 000 000, 6.571},
  {10 000 000 000 000, 8.534}, {100 000 000 000 000, 11.096},
  {1 000 000 000 000 000, 14.031}, {10 000 000 000 000 000, 18.159},
  {100 000 000 000 000 000, 23.143}, {1 000 000 000 000 000 000, 28.479},
  {10 000 000 000 000 000 000, 35.073}, {100 000 000 000 000 000 000, 43.327},
  {1 000 000 000 000 000 000 000, 54.44}, {10 000 000 000 000 000 000 000, 68.642},
  {100 000 000 000 000 000 000 000, 86.6}, {1 000 000 000 000 000 000 000 000, 107.593},
  {10 000 000 000 000 000 000 000 000, 133.439}, {100 000 000 000 000 000 000 000 000, 164.157}}

In[2]:= ListLogLinearPlot[alphaGourdon, Filling → Bottom, Joined → True]

Out[2]=
```



```
(* alpha is a tuning factor that balances the compuation of the
 easy special leaves (A + C formulas) and the hard special leaves
 (D formula). The formula below is used in the file src/util.cpp
 to calculate a fast alpha factor for the computation of pi(x). *)

In[3]:= NonlinearModelFit[alphaGourdon, a (Log[x])^3 + b (Log[x])^2 + c Log[x] + d, {a, b, c, d}, x]

Out[3]= FittedModel[ -183.836 + 16.5791 Log[x] - 0.495545 ≪1≫^2 + 0.00526934 Log[x]^3 ]
```