ArŶmα Labs
Decoding the past, Encoding the future

ROBYN UNDER THE HOOD – Hill Transformation

In this article we will deep dive into how Hill Function is applied in Robyn.

What is Hill Function?

Did you know that the Hill function used for transforming media variables to capture diminishing effect has its origins in Biochemistry and pharmacology?

The equation was formulated by Archibald Hill in 1910 to describe oxygen binding to Haemoglobin.

The formula:

$$SCurve(x) = c \cdot \frac{x^{\alpha}}{(x^{\alpha} + \gamma^{\alpha})}$$

$c =$ coefficient obtained from regression, max/saturation
$x =$ level of (ad–stocked) impressions, exposures
$\alpha > 0$, shape parmater
$\gamma > 0$, inflection parameter

The underlying principle for advertisement is that each additional unit of GRP or impressions increases the KPI (Sales, Conversion etc) only to a certain level and beyond that the effect of advertisement starts diminishing. Hill Function is used to capture this diminishing effect.

Hill Function has two hyper parameters – alpha (α) and gamma (ϒ). Alpha is the shape parameter which controls the shape of the curve. The larger the alpha, the more S-shaped the curve is. For smaller alpha values, the curve is C-shaped. Refer to image below (Image 1) to understand how the curve shape changes with different values of alpha.

Notice in the left side chart, how the curve is S-shaped for alpha = 2 and alpha = 3 and more C-shaped for alpha = 0.1 and alpha = 0.5.

Gamma controls the inflection point of the curve. The higher the value of the gamma, the later the inflection point is on the curve. Refer to image below (Image 1) to understand how the curve saturation changes with different value of gamma. Notice, in the right chart that for gamma = 0.1, the curve saturates the fastest and for gamma = 0.9, the curve is almost linear.

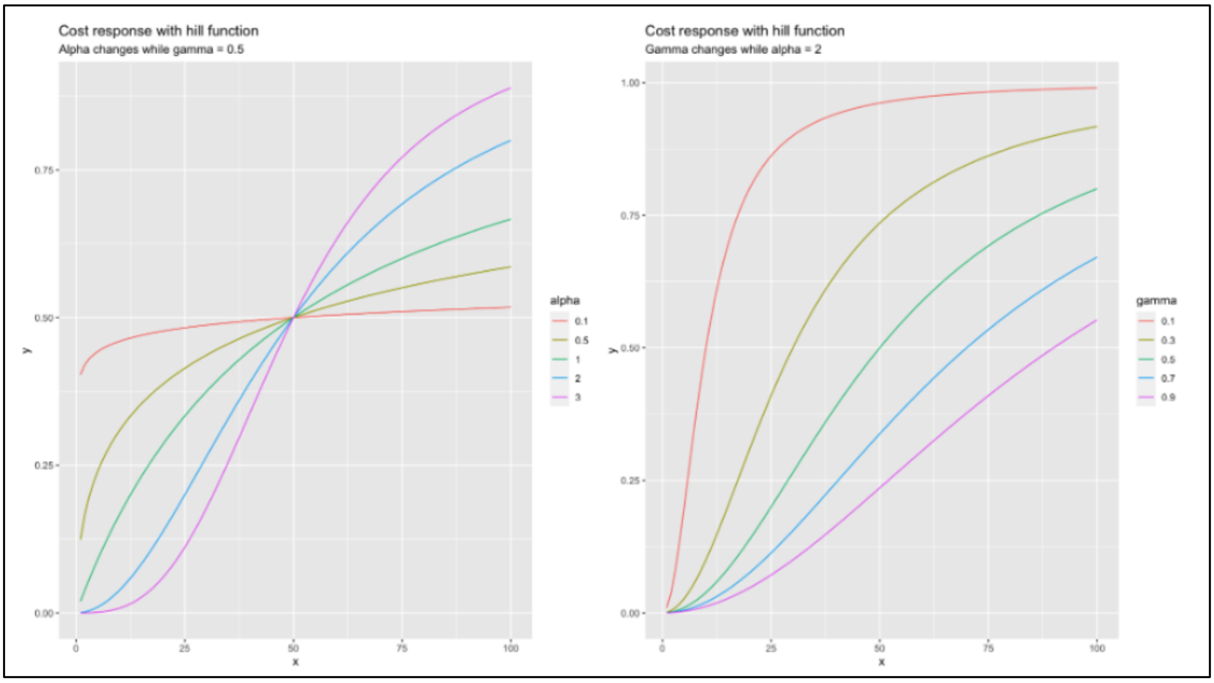

Image 1; Source: [Robyn]

The hill function is applied after the adstock function is applied on the x series (could be TV, Facebook spends etc.). Refer to the modeling equation in Image 2. Xdecay is arrived at after performing adstock (geometric or Weibull on the series). Then, hill function is applied on the xdecay.

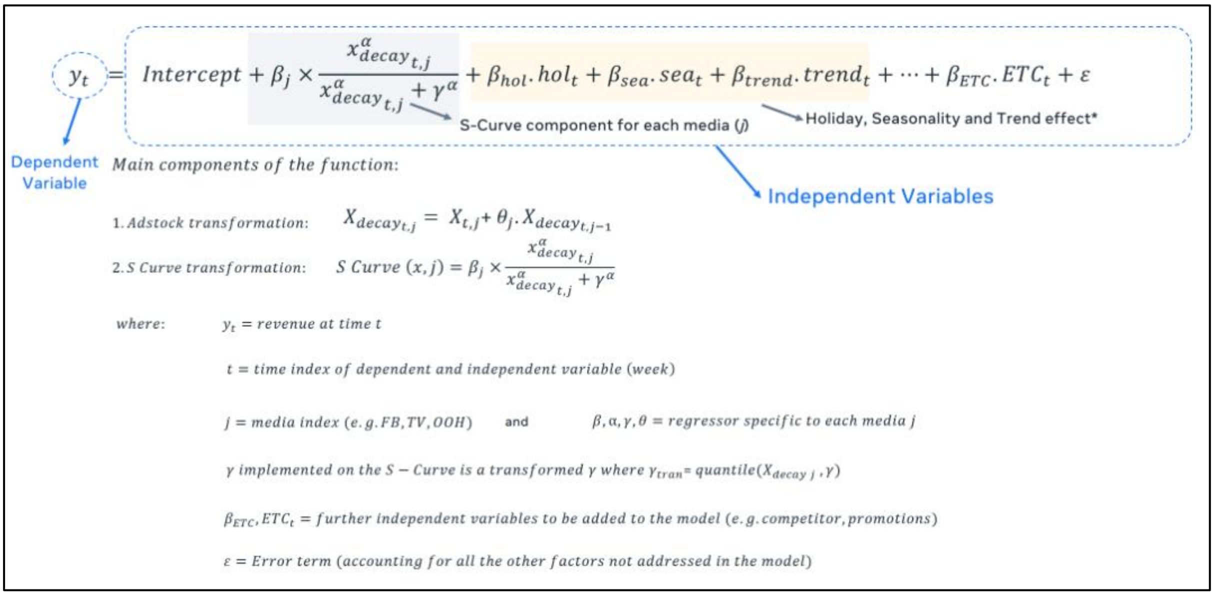

Image 2; Source: [Robyn]

The following code performs hill transformation in Robyn. (Code Snippet 1)

```r
saturation_hill <- function(x, alpha, gamma, x_marginal = NULL) {
  inflexion <- c(range(x) %*% c(1 - gamma, gamma)) # linear interpolation by dot product
  if (is.null(x_marginal)) {
    x_scurve <- x**alpha / (x**alpha + inflexion**alpha) # plot(x_scurve) summary(x_scurve)
  } else {
    x_scurve <- x_marginal**alpha / (x_marginal**alpha + inflexion**alpha)
  }
  return(x_scurve)
}
```

Code Snippet 1, Source: Robyn

Important thing to note is that the gamma used in S-curve formula is a transformed gamma (inflexion). This transformed gamma (inflexion point) is arrived at by first understanding what the minimum and maximum value, a series has.

Let's drill down on how hill function is applied in Robyn:

1. Let's say x is TV Spends. TV Spends for 208 weeks represented below.

```
> x
  [1]  167687.6  214600.9       0.0  625877.3       0.0  249189.8    5152.9   22141.7       0.0
 [10]       0.0  123838.7       0.0  497073.3       0.0  311672.2       0.0  148293.3       0.0
 [19]       0.0       0.0       0.0   83450.5       0.0       0.0       0.0       0.0       0.0
 [28]       0.0       0.0       0.0  153542.9       0.0   75907.6       0.0   14253.1       0.0
 [37]  211351.9       0.0  214730.1       0.0  179933.7  159463.8       0.0       0.0       0.0
 [46] 1084024.8       0.0  610420.1       0.0  704960.9       0.0  392432.9   52177.9       0.0
 [55]  106908.0       0.0       0.0  807907.6  307907.6       0.0   44349.7       0.0  301837.2
 [64]       0.0       0.0       0.0       0.0       0.0   77706.6       0.0       0.0       0.0
 [73]       0.0   56017.9       0.0  274671.9       0.0       0.0       0.0       0.0   68441.0
 [82]       0.0  271947.6       0.0   34205.7       0.0   73978.5       0.0   85386.9       0.0
 [91]  255352.9       0.0  884168.4       0.0  366519.5       0.0  627478.8  834783.4  334783.4
[100]       0.0  332703.8   34839.2  158617.6       0.0  414205.0       0.0  216926.6       0.0
[109]  478117.9  678117.9       0.0       0.0  293169.3       0.0  188769.0       0.0   68739.1
[118]       0.0   92670.8       0.0  398401.9       0.0  148078.8       0.0       0.0       0.0
[127]       0.0       0.0       0.0       0.0    7103.3       0.0       0.0   62360.1       0.0
[136]       0.0  109664.6       0.0   66418.4   72966.7       0.0   97811.2       0.0  461723.4
[145]       0.0   87460.6       0.0       0.0    6659.5 1175275.8       0.0  333671.4       0.0
[154]  189674.8       0.0  430131.9       0.0  401439.0       0.0       0.0       0.0 1185349.3
[163]  385349.3       0.0   86879.8       0.0  237573.0       0.0   95421.3       0.0  135907.2
[172]       0.0   41177.4       0.0   26125.4       0.0  359173.6       0.0   36588.0       0.0
[181]   27719.2       0.0  219302.9       0.0       0.0       0.0       0.0   46203.6       0.0
[190]   39527.7       0.0   90371.3       0.0   97574.8       0.0       0.0       0.0    1698.1
[199]  369508.5       0.0  208627.1       0.0  144479.6       0.0  154917.6   21982.5   22453.0
[208]       0.0
```

2. Let’s apply geometric adstock on TV Spends. For simplicity purpose, let’s assume that the theta = 0.4. Then x_decay will be:

```
> x_decay
  [1]  167687.6000  281675.9400  112670.3760  670945.4504  268378.1802  356541.0721  147769.3288
  [8]   81249.4315   32499.7726   12999.9090  129038.6636   51615.4654  517719.4862  207087.7945
 [15]  394507.3178  157802.9271  211414.4708   84565.7883   33826.3153   13530.5261    5412.2105
 [22]   85615.3842   34246.1537   13698.4615    5479.3846    2191.7538     876.7015     350.6806
 [29]     140.2722      56.1089  153565.3436   61426.1374  100478.0550   40191.2220   30329.5888
 [36]   12131.8355  216204.6342   86481.8537  249322.8415   99729.1366  219825.3546  247393.9419
 [43]   98957.5767   39583.0307   15833.2123 1090358.0849  436143.2340  784877.3936  313950.9574
 [50]  830541.2830  332216.5132  525319.5053  262305.7021  104922.2808  148876.9123   59550.7649
 [57]   23820.3060  817435.7224  634881.8890  253952.7556  145930.8022   58372.3209  325186.1284
 [64]  130074.4513   52029.7805   20811.9122    8324.7649    3329.9060   79038.5624   31615.4250
 [71]   12646.1700    5058.4680    2023.3872   56827.2549   22730.9020  283764.2608  113505.7043
 [78]   45402.2817   18160.9127    7264.3651   71346.7460   28538.6984  283363.0794  113345.2317
 [85]   79543.7927   31817.5171   86705.5068   34682.2027   99259.7811   39703.9124  271234.4650
 [92]  108493.7860  927565.9144  371026.3658  514930.0463  205972.0185  709867.6074 1118730.4430
 [99]  782275.5772  312910.2309  457867.8923  217986.3569  245812.1428   98324.8571  453534.9428
[106]  181413.9771  289492.1909  115796.8763  524436.6505  887892.5602  355157.0241  142062.8096
[113]  349994.4239  139997.7695  244768.1078   97907.2431  107901.9973   43160.7989  109935.1196
[120]   43974.0478  415991.5191  166396.6077  214637.4431   85854.9772   34341.9909   13736.7964
[127]    5494.7185    2197.8874     879.1550     351.6620    7243.9648    2897.5859    1159.0344
[134]   62823.7137   25129.4855   10051.7942  113685.3177   45474.1271   84608.0508  106809.9203
[141]   42723.9681  114900.7873   45960.3149  480107.5260  192043.0104  164277.8042   65711.1217
[148]   26284.4487   17173.2795 1182145.1118  472858.0447  522814.6179  209125.8472  273325.1389
[155]  109330.0555  473863.9222  189545.5689  477257.2276  190902.8910   76361.1564   30544.4626
[162] 1197567.0850  864376.1340  345750.4536  225179.9814   90071.9926  273601.7970  109440.7188
[169]  139197.5875   55679.0350  158178.8140   63271.5256   66486.0102   26594.4041   36763.1616
[176]   14705.2647  365055.7059  146022.2823   94996.9129   37998.7652   42918.7061   17167.4824
[183]  226169.8930   90467.9572   36187.1829   14474.8732    5789.9493   48519.5797   19407.8319
[190]   47290.8328   18916.3331   97937.8332   39175.1333  113244.8533   45297.9413   18119.1765
[197]    7247.6706    4597.1682  371347.3673  148538.9469  268042.6788  107217.0715  187366.4286
[204]   74946.5714  184896.2286   95940.9914   60829.3966   24331.7586
>
```

3. Compute the inflexion point to run the hill function.

   a. Let’s compute the minimum and maximum values of the x_decay series.

```
> range(x_decay)
[1]       56.1089 1197567.0850
>
```

   Here the min value is 56.1 and the max value is 1,197,567.

   b. Perform a dot product of min and max value with (1-gamma) and gamma values. Assume gamma = 0.3.

```
> inflexion <- c(range(x_decay) %*% c(1 - gamma, gamma)) # linear interpolation by dot product
> print(inflexion)
[1] 359309.4
>
```

The inflexion point will be 359,309.4. The inflexion point is computed based on how much spends or impressions have occurred in the time period for which the model is built.

Another approach mentioned in Robyn to compute the inflexion point is to use quantile function. The quantile function can be applied on the series x_decay using the gamma value as the probability.

Inflexion = quantile(x_decay, 0.3)

ArŶmα Labs
Decoding the past, Encoding the future

4. Run the hill function formula on x_decay using the inflexion point value. (Assume gamma = 0.3 and alpha = 0.5).

   This results in x_scurve which is a transformed series on which adstock and diminishing returns transformations are applied.

```
> x_scurve <- x_decay**alpha / (x_decay**alpha + inflexion**alpha) # plot(x_scurve) summary(x_scurve)
> x_scurve
  [1] 0.40587595 0.46960918 0.35896512 0.57743485 0.46359121 0.49903320 0.39072501 0.32227639
  [9] 0.23121286 0.15981291 0.37471639 0.27484438 0.54552974 0.43155232 0.51167960 0.39857205
 [17] 0.43409037 0.32666077 0.23478756 0.16251708 0.10931442 0.32801880 0.23589746 0.16335827
 [25] 0.10991638 0.07244389 0.04707087 0.03029434 0.01937556 0.01234207 0.39531402 0.29252044
 [33] 0.34589735 0.25062765 0.22512777 0.15522752 0.43684431 0.32912955 0.45444726 0.34505148
 [41] 0.43888826 0.45348466 0.34417442 0.24919847 0.17349789 0.63530384 0.52420442 0.59644442
 [49] 0.48313801 0.60323122 0.49020164 0.54733567 0.46074679 0.35080970 0.39161421 0.28932238
 [57] 0.20475718 0.60132625 0.57068036 0.45672906 0.38923579 0.28727186 0.48752932 0.37565347
 [65] 0.27564180 0.19398380 0.13210490 0.08781425 0.31927100 0.22877003 0.15796944 0.10606698
 [73] 0.06980391 0.28453347 0.20097218 0.47052920 0.35981542 0.26224923 0.18355336 0.12448779
 [81] 0.30824945 0.21986360 0.47035297 0.35965248 0.31996382 0.22933262 0.32941475 0.23703966
 [89] 0.34451863 0.24948384 0.46490776 0.35463076 0.61637499 0.50401107 0.54485995 0.43088979
 [97] 0.58429918 0.63827454 0.59604474 0.48272344 0.53026257 0.43785410 0.45268992 0.34345086
[105] 0.52907821 0.41539631 0.47301929 0.36212032 0.54712729 0.61119389 0.49854702 0.38604747
[113] 0.49671672 0.38431364 0.45216269 0.34297113 0.35400511 0.25738101 0.35614244 0.25916901
[121] 0.51830195 0.40494445 0.43594965 0.32832687 0.23614941 0.16354933 0.11005316 0.07253784
[129] 0.04713358 0.03033542 0.12433462 0.08240175 0.05374312 0.29485378 0.20914762 0.14329154
[137] 0.35999755 0.26240222 0.32671572 0.35284282 0.25641004 0.36122358 0.26343268 0.53616501
[145] 0.42232598 0.40340139 0.29954660 0.21288819 0.17940048 0.64461489 0.53427259 0.54674349
[153] 0.43275395 0.46586297 0.35550992 0.53453696 0.42073005 0.53542452 0.42159980 0.31553782
[161] 0.22574413 0.64609813 0.60799977 0.49519183 0.44185379 0.33363560 0.46598884 0.35562583
[169] 0.38363571 0.28246034 0.39885724 0.29559270 0.30077793 0.21387207 0.24234904 0.16826285
[177] 0.50198325 0.38931029 0.33957926 0.24539688 0.25684382 0.17937563 0.44239475 0.33412338
[185] 0.24090223 0.16716074 0.11264237 0.26872359 0.18858151 0.26621073 0.18662681 0.34300633
[193] 0.24823072 0.35955047 0.26202672 0.18338103 0.12436246 0.10161827 0.50411916 0.39134351
[201] 0.46343568 0.35327733 0.41932162 0.31352210 0.41770675 0.34068901 0.29151130 0.20649220
```

So, this sums up how Hill Function is applied in Robyn.

References:

1. https://facebookexperimental.github.io/Robyn/docs/features/
2. https://facebookexperimental.github.io/Robyn/docs/analysts-guide-to-MMM/
3. https://github.com/facebookexperimental/Robyn/blob/main/R/R/transformation.R