Geometric complexity comparison of Holling type II and Michaelis-Menten formulations.

The Holling type II and Michaelis - Menten functional forms represent different parameterizations of the same functional form. Therefore, they should have the same structural complexity when their parameter values are equivalently constrained.

Preliminaries

```
Clear["Global`*"]
SetDirectory[FileNameJoin[{NotebookDirectory[], "../figs/"}]] ;
```

Define Fisher information matrix

```
In[∘]:= fisher[fun_, pars_] := -D[Apply[fun, pars], {pars, 2}]
```

Define functional responses - number of prey eaten

In[∘]:=
$$H2 = \frac{a\,N}{1 + a\,h\,N}\,P\,T;$$

$$MM = \frac{\alpha\,N}{\beta + N}\,P\,T;$$

Experimental Design

```
In[∘]:= subs = {T → 1, P → 1};
Nvals = Table[Fibonacci[i], {i, 4, 8}]
Pvals = Table[Fibonacci[i], {i, 2, 2}]

Nprobs = ConstantArray[1 / Length[Nvals], Length[Nvals]];
NDistE = EmpiricalDistribution[Nprobs → Nvals];
Pprobs = ConstantArray[1 / Length[Pvals], Length[Pvals]];
PDistE = EmpiricalDistribution[Pprobs → Pvals];

Out[∘]= {3, 5, 8, 13, 21}

Out[∘]= {1}
```

Define Expected Fisher information matrix given experimental design

Note that we remove the distribution on predator abundances here (cf. GeomComp_Compute.nb) as we assume P = {1} for simplicity

```
In[•]:= EFisher[FisherMatrix_, func_, NDistE_] :=
         Expectation[
          Expectation[
           FisherMatrix,
           x ≈ PoissonDistribution[func]],
          {N ≈ NDistE}]
```

Poisson log Likelihood

```
In[•]:= PoislL[func_] := -n func + Log[func] x /. {n → 1}
```

Determinant - Holling Type II

```
In[•]:= lLH2Poiss[a_, h_] := PoislL[H2]
        FisherH2Poiss = fisher[lLH2Poiss, {a, h}];
        % // FullSimplify // MatrixForm
```

Out[•]//MatrixForm=

$$\begin{pmatrix} \frac{x+3\,a\,h\,x\,N+2\,a^2\,h\,(-P\,T+h\,x)\,N^2}{a^2\,(1+a\,h\,N)^3} & \frac{N\,(x-2\,a\,P\,T\,N+a\,h\,x\,N)}{(1+a\,h\,N)^3} \\ \frac{N\,(x-2\,a\,P\,T\,N+a\,h\,x\,N)}{(1+a\,h\,N)^3} & -\frac{a^2\,N^2\,(x-2\,a\,P\,T\,N+a\,h\,x\,N)}{(1+a\,h\,N)^3} \end{pmatrix}$$

```
In[•]:= DetH2 = Det[EFisher[FisherH2Poiss /. subs, H2 /. subs, NDistE]];
```

Determinant - Michaelis-Menten

```
In[•]:= lLMMPoiss[α_, β_] := PoislL[MM]
        FisherMMPoiss = fisher[lLMMPoiss, {α, β}];
        % // FullSimplify // MatrixForm
```

Out[•]//MatrixForm=

$$\begin{pmatrix} \frac{x}{\alpha^2} & -\frac{P\,T\,N}{(\beta+N)^2} \\ -\frac{P\,T\,N}{(\beta+N)^2} & \frac{2\,P\,T\,\alpha\,N-x\,(\beta+N)}{(\beta+N)^3} \end{pmatrix}$$

```
In[•]:= DetMM = Det[EFisher[FisherMMPoiss /. subs, MM /. subs, NDistE]];
```

Integrate square root of Expected Fisher Information Matrix over parameters, and take natural log

Note that the limits only become an issue when we don't explore the full parameter space. If we let a and b go from zero to infinity (which in theory they could, even if that makes the numeric integral blow up in Mathematica....), then α and β could go from 0 to infinity too and the result should continue to match.
For the H2 model, we restrict a to be less than amax, and h to be less than hmax.
The parameter range of MM must be restricted to equivalent parameter space.

Algebraically, α = 1/h and β = 1/(ah) (also $\beta = \alpha$ / a). The bounds on α hence only depend on the bounds on h, and thus are [1/hmax, 1/hmin] = [1/hmax, ∞]. On the other hand, when performing the integration the limits for β depends on the value of α. Therefore, for a fixed value of α, the limits of integration over β vary from [α/β, ∞]. This can be achieved either by including α within the limit of integration, which can be accomplished with calculus. However, NIntegrate[] isn't as clever, so we instead state that the values of the integrand Sqrt[DetMM] only contribute non-zero values to the integral within the permissible range of the integration.

In[∘]:=
```
amax = 10;
hmax = 10;

ParmRange = {
   {a, 0, amax},
   {h, 0, hmax}};
MMParmRange = {
   {α, 1 / hmax, Infinity} ,
   {β, 1 / (amax * hmax) , Infinity}};

accgoal = 6;
precgoal = 6;

NIntH2 = Log[NIntegrate[
   Sqrt[DetH2],
   ParmRange[[1]], ParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal]]
NIntMM = Log[NIntegrate[
   Sqrt[DetMM] Boole[α / amax < β], (* Note additional constraint here *)
   MMParmRange[[1]], MMParmRange[[2]],
   AccuracyGoal → accgoal,
   PrecisionGoal → precgoal]]
```

Out[∘]= 3.36559

Out[∘]= 3.36559

Visualization

We additionally limit the parameter range of both models such that:

the maximum number of prey eaten does not exceed the number of prey available in the highest prey abundance treatment

and that

the minimum number of prey eaten is not less than 1 prey individual

Note that allowable parameter space is dependent on species abundances (i.e. treatment levels). For example, for H2, h must satisfy (1/Nmax - 1/aN) < h < (1 - 1/aN). Thus, in the following visualization, we (arbitrarily) pick an exemplar prey abundance (Nfocal). By contrast, all treatment levels are "summed over" when calculating $\int_\Theta \sqrt{\det I(\Theta)}\, d\Theta$.

In[∘]:=

```
H2f[a_, h_] := (a N)/(1 + a h N) P T;
MMf[α_, β_] := (α N)/(β + N) P T;
H2 = H2f[a, h];
MM = MMf[α, β];

Nvals = Table[Fibonacci[i], {i, 4, 8}];
P = 1;
T = 1;

parmsH2 = {a → 6, h → 0.06};
parmsMM = {α → 1 / h, β → 1 / (a h)} /. parmsH2;
parms = Join[parmsH2, parmsMM];

TitleFontSize = 14;
LabelFontSize = 12;

(* H2 Functional response *)
frH2 = Plot[H2 /. parms,
   {N, 0, Max[Nvals]},
   PlotRange → {{0, 1.05 * Max[Nvals]}, {0, 1.1 * Max[Nvals]}},
   Frame → True,
   FrameLabel →
    {Style["Prey abundance", LabelFontSize], Style["Prey eaten", LabelFontSize]},
   PlotLabel → Style["Holling Type II", TitleFontSize],
   PlotStyle → Black];

(* MM functional response  *)
frMM = Plot[MM /. parms,
   {N, 0, Max[Nvals]},
    PlotRange → {{0, 1.05 * Max[Nvals]}, {0, 1.1 * Max[Nvals]}},
   Frame → True,
   FrameLabel →
    {Style["Prey abundance", LabelFontSize], Style["Prey eaten", LabelFontSize]},
   PlotLabel → Style["Michaelis-Menten", TitleFontSize],
   PlotStyle → Black
  ];
```

```
(* Functional response region (for both)*)
frReg = RegionPlot[
   {
    {F < (1 / Max[Nvals] P T) * N} ,
    {F > (H2f[1000, 1 / Max[Nvals] P T ] /. N → Max[Nvals])}
   },
   {N, 0, 1.05 * Max[Nvals]}, {F, 0, 1.1 * Max[Nvals]},
   BoundaryStyle → None,
   PlotStyle →
    {{ColorData[97, 1], Opacity[0.5]}, {ColorData[97, 2], Opacity[0.5]}},
   Frame → False];

(* Experimental design points *)
frDes = ListPlot[
   Append[Transpose@{Nvals, Table[Max[Nvals], Length[Nvals]]}, {Max[Nvals], 1}],
   PlotStyle → ColorData["M10DefaultDensityGradient", "ColorFunction"][0.1],
   PlotMarkers → "★"
  ];

(* Annotations *)
ascale = 0.08 * Max[Nvals];
hscale = 0.25 * Max[Nvals];

aText = Text["a", {ascale * 1.2, a * ascale /. parms}];
hText = Text["1/h", {hscale * 1.2, 1.01 * 1 / h /. parms}];
αText = Text["α", {hscale * 1.2, 1.01 * α /. parms}];
βText = Text["β", {1.2 * β /. parms, 0.35 * α /. parms}];

(* Combine for functional response H2 *)
pfrH2 = Show[frH2, frDes,
   Graphics[{Dashed, Gray,
     Line[{{0, 0}, {ascale, a * ascale /. parms}}]}],
   Graphics[{Dashed, Gray,
     Line[{{0, 1 / h} /. parms, {hscale, 1 / h} /. parms}]}],
   Graphics[aText],
   Graphics[hText]
  ];

(* Combine for functional response MM *)
pfrMM = Show[frMM, frDes,
   Graphics[{Dashed, Gray,
     Line[{{β /. parms, 0}, {β /. parms, 0.35 * α /. parms}}]}],
   Graphics[{Dashed, Gray,
```

```
      Line[{{0, α} /. parms, {hscale, α} /. parms}}}],
    Graphics[βText],
    Graphics[αText]
   ];

amax = 0.5;
amin = 0.2;
hmax = 10;
hmin = 1;

pRegParH2 = RegionPlot[
    amin < a && a < amax &&
     hmin < h && h < hmax ,
    {a, 0, amax * 1.05}, {h, 0, hmax * 1.05},
    Frame → True,
    FrameLabel → {Style[a, LabelFontSize], Style[h, LabelFontSize]},
    PlotRangePadding → None,
    BoundaryStyle → GrayLevel[0.3],
    PlotStyle → GrayLevel[0.6],
    Epilog → {
      {Text["𝒟H2", {0.45, 1.8}]}
     }];

αmin = 1 / hmax;
αmax = 1 / hmin;
βmin = 1 / (hmax * amax);
βmax = 1 / (hmin * amin);
pRegParMM = RegionPlot[
    {
     (* Within specified restrictions *)
     {
      αmin < α && α < αmax &&
       βmin < β && β < βmax },
     {
      α / amax > β }
    },

    {β, 0, 1.05 * βmax}, {α, 0, 1.05 * αmax},
    Frame → True,
    FrameLabel → {Style[β, LabelFontSize], Style[α, LabelFontSize]},
    PlotRangePadding → None,
    BoundaryStyle → GrayLevel[0.3],
    PlotStyle → {
      GrayLevel[0.6],
```

```
      {GrayLevel[0.1], HatchFilling[Pi / 4, 0.1]}
     },
    Epilog → {
      {Arrow[{{2.2, 0.85}, {1.3, 0.9}}]},
      {Text["α/amax > β", {3.1, 0.85}]},
      {Text["𝒟MM", {4.5, 0.18}]}
     }];

(* H2 restricted by F *)
MinFH2[a__?NumericQ, h__?NumericQ] = Min[H2 /. {N → Max[Nvals]}];
MaxFH2[a__?NumericQ, h__?NumericQ] = Max[H2 /. {N → Nvals}];

pRegFH2 = ContourPlot[Sqrt[DetH2], {a, 0, 5}, {h, 0, 1},
    RegionFunction → Function[{a, h}, MinFH2[a, h] > 1 && MaxFH2[a, h] < Max[Nvals]],
    (*ColorFunction→GrayLevel,*)
    PlotRange → All,
    Contours → 100,
    Frame → True,
    FrameLabel → {Style[a, LabelFontSize], Style[h, LabelFontSize]},
    PlotRangePadding → None,
    Epilog → {
      {White, Arrow[{{1.2, 0.83}, {0.7, 0.93}}]},
      {White, Text["E[F(Nmax) PT] = 1", {1.85, 0.78}]},
      {White, Arrow[{{4, 0.17}, {4.2, 0.04}}]},
      {White, Text["E[F(N) PT] = Nmax", {3.5, 0.22}]}
     }
   ];

(* MM restricted by F *)
MinFMM[α__?NumericQ, β__?NumericQ] = Min[MM /. {N → Max[Nvals]}];
MaxFMM[α__?NumericQ, β__?NumericQ] = Max[MM /. {N → Nvals}];

pRegFMM = ContourPlot[Sqrt[DetMM], {β, 0, 40}, {α, 0, 40},
    RegionFunction → Function[{β, α}, MinFMM[α, β] > 1 && MaxFMM[α, β] < Max[Nvals]],
    (*ColorFunction→GrayLevel,*)
    PlotRange → All,
    Contours → 20,
    Frame → True,
    FrameLabel → {Style[β, LabelFontSize], Style[α, LabelFontSize]},
    PlotRangePadding → None,
    Epilog → {
      {White, Arrow[{{20.5, 32}, {16, 37}}]},
      {White, Text["E[F(N) PT] = Nmax", {26, 30}]},
      {White, Arrow[{{30, 8}, {31, 3.1}}]},
```

```
     {White, Text["𝔼[F(N_max) PT] = 1", {28, 10}]}
    }];

(* Combine and export all plots *)
AllPlots = GraphicsGrid[{
    {pfrH2, pfrMM},
    {pRegParH2, pRegParMM},
    {pRegFH2, pRegFMM}}, Spacings → {0, -10}];
Export["GeomComp_H2vMM.pdf", Show[AllPlots, ImageSize → 6 * 72]];
```