```
(* PolySem - Demo  *)
FPrint[a_] := (Print[a]; a);
MyRound[L_List, dx_] := Map[MyRound[#, dx] &, L];
MyRound[x_, dx_] := Round[x, dx] /; NumberQ[x];
MyRound[x_Symbol, dx_] := x;
MyRound[x_String, dx_] := x;
MyRound[a_ → x_, dx_] := a → MyRound[x, dx];
MyRound[Var[x_], dx_] := Var[x];
MyRound[Cov[x_, y_], dx_] := Cov[x, y];
Options[PolySem] = {verbose → False, doNMini → False, userConstr → {},
  usePI → False, setVarPos → False, ConstrainLatent1To0 → False,
  NDObs → {}, doML → False, Level → 2, Include1 → True, WarmStart → <||>};
PolySem[data_, eqs_, latent0_, latent1_, errorvars_,
  parameters_, prec_List, OptionsPattern[]] :=
 Module[{dat = Rest[data], datT, observed = First[data], S, Z, Observed, i,
   j, Tr2, Wmat2, Fwls, FWLS, FWLSm, FWLSm0, vect, index2, Flatten1, AsList,
   ScalarQ, COV, COV3, COV4, headlist, getAllVariables, res, Print0, W, WI,
   constr = {}, vv, F1, F2, F3, F4, S1 = 0, S2 = 0, S3 = 0, S4 = 0, Z1 = 0, Z2 = 0, Z3 = 0,
   Z4 = 0, f, f2, Optimizer, options, factor1, TakeByIndices, MTakeByIndices, IL,
   val0, val01, val02, val03, val04, FULS1 = 0, FULS2 = 0, FULS3 = 0, FULS4 = 0},
  latent = Join[latent0, latent1]; mean0vars = Join[errorvars, latent0];
  Observed = Expand[observed //. eqs];
  AsList[a_] := Table[a[[i]], {i, 1, Length[a]}];
  Flatten1[L_] := If[MemberQ[L, {}], {}, Flatten[L]];
  IsProductOfIDPMean0Vars[a_^n_] :=
   Flatten[Table[IsProductOfIDPMean0Vars[a], n]] /; IntegerQ[n] && n > 0;
  IsProductOfIDPMean0Vars[a_] := Module[{L, i, n = Length[a], vlist},
    If[MemberQ[mean0vars, a], {a},
     If[Not[Head[a] === Times], {},
      Flatten1[Map[IsProductOfIDPMean0Vars, AsList[a]]]]]];
  Print0[a_] := Print[MyRound[a, 0.000001]];
  datT = Transpose[dat];
  S1 = Map[Mean, datT];
  S2 = Covariance[dat];
If[OptionValue[Level] ≥ 3, S3 = Table[Mean[datT[[i]] * datT[[j]] * datT[[k]] -
       Mean[datT[[i]]] * Mean[datT[[j]]] * Mean[datT[[k]], {i, 1, Length[observed]},
      {j, 1, Length[observed]}, {k, 1, Length[observed]}]];
If[OptionValue[Level] ≥ 4, S4 = Table[Mean[datT[[i]] * datT[[j]] * datT[[k]] * datT[[l]] -
       Mean[datT[[i]]] * Mean[datT[[j]]] * Mean[datT[[k]] * Mean[datT[[l]],
      {i, 1, Length[observed]}, {j, 1, Length[observed]},
      {k, 1, Length[observed]}, {l, 1, Length[observed]}]];
  headlist = {Or, And, Equal, Unequal, Less, LessEqual,
    Greater, GreaterEqual, Inequality};
  getAllVariables[f_?NumericQ] := {};
getAllVariables[{}] := {};
getAllVariables[a_ → b_] := Union[getAllVariables[a], getAllVariables[b]];
getAllVariables[t_] /; MemberQ[headlist, t] := {};
```

```
getAllVariables[ll_List] :=
   Union[Flatten[Union[Map[getAllVariables[#] &, ll]]]];
getAllVariables[Derivative[n_Integer][f_][arg__]] := getAllVariables[{arg}];
getAllVariables[f_Symbol[arg__]] := Union[{Module[{fvars},
      If[MemberQ[Attributes[f], NumericFunction] || MemberQ[headlist, f],
       fvars = getAllVariables[{arg}], (*else*)fvars = f[arg]];
      fvars]}];
getAllVariables[other_] := {other};
GetAllVariables[a_] := Union[Flatten[getAllVariables[a]]];
Cov[a_ + b_, c_] := Cov[a, c] + Cov[b, c];
Cov[c_, a_ + b_] := Cov[a, c] + Cov[b, c];
Cov[s_, b_] := 0 /; ScalarQ[s];
Cov[b_, s_] := 0 /; ScalarQ[s];
Cov[s_ * a_, b_] := s * Cov[a, b] /; ScalarQ[s];
Cov[a_, s_ * b_] := s * Cov[a, b] /; ScalarQ[s];
Cov[a_, b_] := Cov[b, a] /; Order[a, b] < 0;
Cov[a_, a_] := Var[a];
Var[a_ + b_] := Var[a] + Var[b] + 2 * Cov[a, b];
Var[s_ * a_] := s^2 * Var[a] /; ScalarQ[s];
ScalarQ[a_] := True /; NumberQ[a] || MemberQ[parameters, a];
ScalarQ[a_ + b_] := ScalarQ[a] && ScalarQ[b];
ScalarQ[a_ * b_] := ScalarQ[a] && ScalarQ[b];
ScalarQ[a_^b_] := ScalarQ[a] && ScalarQ[b];
P2n[{}] := {}; P2n[{a_, b_}] := {{Sort[{a, b}]}};
P2n[L_List] := {} /; OddQ[Length[L]];
P2n[L_List] := Module[{n = Length[L],
      a = First[L], R = Rest[L], i, res = {}, P}, Do[P = Sort[{a, R[[i]]}];
      Map[AppendTo[res, Join[{P}, #]] &,
       P2n[Join[Take[R, i - 1], Take[R, i + 1 - n]]]], {i, 1, n - 1}];
     res] /; EvenQ[Length[L]];
IsserList[L_List] :=
   Module[{p, ij}, Sum[Product[Cov[First[ij], Last[ij]], {ij, p}], {p, P2n[L]}]];
  EV[a_ + b_] := EV[a] + EV[b];
  EV[a_ * b_] := EV[Expand[a * b]] /; Head[Expand[a * b]] == Plus;
  EV[(a_ + b_)^n_Integer] := EV[Expand[(a + b)^n]];
  EV[c_ * a_] := c * EV[a] /; ScalarQ[c];
  EV[c_^n_] := c^n /; ScalarQ[c];
  Ev[c_^n_] := c^n /; ScalarQ[c];
  Ev[c_] := c /; ScalarQ[c];
  EV[a_] := a /; NumberQ[a];
  (*Fehler unkorreliert*)
  EV[a_Symbol] := If[TrueQ[ScalarQ[c]], c,
    If[TrueQ[MemberQ[errorvars, a] || MemberQ[latent0, a]], 0, Ev[a]]];
  EV[a_^2] := Var[a] /; MemberQ[errorvars, a];
  EV[a_^n_Integer] := 0 /; OddQ[n] && MemberQ[errorvars, a];
  EV[a_ * b_] := 0 /; (MemberQ[errorvars, a] && MemberQ[errorvars, b]);
  EV[a_ * b_] := 0 /; MemberQ[errorvars, a] && MemberQ[latent, b];
```

```
  EV[b_ * a_] := 0 /; MemberQ[errorvars, a] && MemberQ[latent, b];
  EV[a_ * b_] := 0 /; MemberQ[errorvars, a] && MemberQ[observed, b];
  EV[b_ * a_] := 0 /; MemberQ[errorvars, a] && MemberQ[observed, b];
  EV[a_ * b_] := Cov[a, b] /; MemberQ[latent, a] && MemberQ[latent, b];
  EV[a_] := 0 /; MemberQ[mean0vars, a];
  COV[a_, b_] := EV[a * b] - EV[a] * EV[b];
  COV3[a_, b_, c_] := EV[Expand[a * b * c]] - EV[a] * EV[b] * EV[c];
  COV4[a_, b_, c_, d_] := EV[Expand[a * b * c * d]] - EV[a] * EV[b] * EV[c] * EV[d];
EV[a_] := IsserList[FlatPowers[IsProductOfIDPMean0Vars[a]]] /;
    Length[IsProductOfIDPMean0Vars[a]] > 0;
Cov[a_, b_] := 0 /; (Not[a === b] && MemberQ[errorvars, a] && MemberQ[errorvars, b]);
  (*Fehler unkorreliert*)
  Cov[a_, b_] := 0 /; (MemberQ[errorvars, a] && MemberQ[latent, b]);
  (*Fehler unkorreliert*)
  Cov[b_, a_] := 0 /; (MemberQ[errorvars, a] && MemberQ[latent, b]);
  (*Fehler unkorreliert*)
  Cov[a_, b_] := 0 /; (MemberQ[errorvars, a] && MemberQ[observed, b]);
  (*Fehler unkorreliert*)
  Cov[b_, a_] := 0 /; (MemberQ[errorvars, a] && MemberQ[observed, b]);
  (*Fehler unkorreliert*)
FlatPowers[L_List] := Module[{i, j, res = {}}, Do[If[
      TrueQ[Head[L[[i]]] == Power], res = Join[res, Table[L[[i]][[1]], {j, 1, L[[i]][[2]]}]],
      res = Join[res, {L[[i]]}]], {i, 1, Length[L]}];
    res];
  Tr2[A_] := Tr[A.A];
  vect[A_] := Module[{ii, jj, res = {}, n = Length@First@A, m = Length@A},
    Do[Do[AppendTo[res, A[[ii]][[jj]]], {ii, 1, jj}], {jj, 1, n}];
    res];
  index2[s_, m_] := Module[{c = 0, ii, jj, res = Null},
    Do[Do[(c = c + 1; If[c == s, res = {ii, jj}]), {ii, 1, jj}], {jj, 1, m}];
    res];
  Wmat2[data0_] := Module[{i, j, k, l, n = Length@data0,
     m = Length@First@data0, dt = Transpose@data0, xm, M, w2, w4, r, u, s, t},
    xm = Map[Mean, dt]; M = m * (m + 1) / 2; (* dimension von vect *)
    w2 = Table[
      1 / n * Apply[Plus, (dt[[i]] - xm[[i]]) * (dt[[j]] - xm[[j]])], {i, 1, m}, {j, 1, m}];
    w4 = Table[1 / n * Apply[Plus, (dt[[i]] - xm[[i]]) * (dt[[j]] - xm[[j]]) *
          (dt[[k]] - xm[[k]]) * (dt[[l]] - xm[[l]])],
      {i, 1, m}, {j, 1, m}, {k, 1, m}, {l, 1, m}];
    u = Table[ w4[[i, j, k, l]] - w2[[i, j]] * w2[[k, l]],
      {i, 1, m}, {j, 1, m}, {k, 1, m}, {l, 1, m}];
    Table[u[[First[index2[s, m]], Last[index2[s, m]], First[index2[t, m]],
      Last[index2[t, m]] ]], {s, 1, M}, {t, 1, M}]
   ];
  W = Wmat2[dat]; res["W"] = W;
  If[OptionValue[usePI], WI = PseudoInverse[W], WI = Inverse[W]];
  Z1 = Expand[Table[EV[Observed[[i]] ], {i, 1, Length[observed]}] ];
```

```
If[OptionValue[verbose] && False, Print[{"Z1", Z1}]];
If[KeyExistsQ[OptionValue[WarmStart], "Z2"],
 Z2 = OptionValue[WarmStart]["Z2"],
 Z2 = Expand[Table[Table[COV[Observed[[i]], Observed[[j]] ],
     {i, 1, Length[observed]}], {j, 1, Length[observed]}] ]];
If[OptionValue[verbose], Print[{"Z2", GetAllVariables[Z2]}]];
If[OptionValue[Level] ≥ 3,
 If[KeyExistsQ[OptionValue[WarmStart], "Z3"],
  Z3 = OptionValue[WarmStart]["Z3"],
  Z3 = Expand[Table[COV3[Observed[[i]], Observed[[j]], Observed[[k]] ], {i, 1, Length[
       observed]}, {j, 1, Length[observed]}, {k, 1, Length[observed]}] ]];
 If[OptionValue[verbose], Print[{"Z3", GetAllVariables[Z3]}]] ];
If[OptionValue[Level] ≥ 4,
 If[KeyExistsQ[OptionValue[WarmStart], "Z4"],
  Z4 = OptionValue[WarmStart]["Z4"],
  Z4 = Expand[Table[COV4[Observed[[i]], Observed[[j]], Observed[[k]], Observed[[l]] ],
     {i, 1, Length[observed]}, {j, 1, Length[observed]},
     {k, 1, Length[observed]}, {l, 1, Length[observed]}] ] ];
 If[OptionValue[verbose], Print[{"Z4", GetAllVariables[Z4]}]] ];
res = <|"n" → Length[dat], "S1" → S1, "Z1" → Z1,
  "S2" → S2, "Z2" → Z2, "S3" → S3, "Z3" → Z3, "S4" → S4, "Z4" → Z4|>;
constr = OptionValue[userConstr];
If[OptionValue[setVarPos], Do[If[Head[vv] == Var, AppendTo[constr, vv > 0]],
  {vv, GetAllVariables[{Z1, Z2, Z3, Z4}]}]];
If[OptionValue[ConstrainLatent1To0],
 Do[AppendTo[constr, 0 == EV[vv /. eqs]], {vv, latent1}]];
res["constr"] = constr;
options = Flatten[{MaxIterations → 150 000,
   If[prec == {}, {WorkingPrecision → Automatic, PrecisionGoal → Automatic},
    { WorkingPrecision → prec[[1]],  PrecisionGoal → prec[[2]] } ]}];
If[OptionValue[doNMini], AppendTo[options, Method → {"NelderMead",
    "ShrinkRatio" → 0.95, "ContractRatio" → 0.95, "ReflectRatio" → 2}]];
Optimizer[of_, vals_] := If[OptionValue[doNMini],
  Apply[NMinimize, Join[{of, Map[First, vals]}, options ] ],
  Apply[FindMinimum, Join[{of, vals}, options]]];
If[OptionValue[Include1],
 FULS1 = 1 / 2 * Apply[Plus, Map[#^2 &, Flatten[S1 - Z1]]];
 factor1 = 1, factor1 = 0];
FULS2 = 1 / 2 * Apply[Plus, Map[#^2 &, Flatten[S2 - Z2]]];
If[OptionValue[Level] ≥ 3,
 FULS3 = 1 / 2 * Apply[Plus, Map[#^2 &, Flatten[S3 - Z3]]]];
If[OptionValue[Level] ≥ 4, FULS4 =
  1 / 2 * Apply[Plus, Map[#^2 &, Flatten[S4 - Z4]]]];
val01 = Map[{#, 0.5} &, GetAllVariables[{FULS1, constr}]];
val02 = Map[{#, 0.5} &, GetAllVariables[{FULS1 + FULS2, constr}]];
val03 = Map[{#, 0.5} &, GetAllVariables[{FULS1 + FULS2 + FULS3, constr}]];
val04 = Map[{#, 0.5} &, GetAllVariables[{FULS1 + FULS2 + FULS3 + FULS4, constr}]];
```

```
If[OptionValue[Include1],
 res["ULS1"] = Optimizer[Prepend[constr, FULS1], val01];
 If[OptionValue[verbose], Print0[{"ULS1", res["ULS1"]}]]];
res["ULS2"] =
 Optimizer[Prepend[constr, factor1 * FULS1 + FULS2 / Length[observed]], val02];
If[OptionValue[verbose], Print0[{"ULS2", res["ULS2"]}]];
If[OptionValue[Level] ≥ 3,
 res["ULS3"] = Optimizer[Prepend[constr, factor1 * FULS1 +
     FULS2 / Length[observed] + FULS3 / Length[observed]^2], val03];
 If[OptionValue[verbose], Print0[{"ULS3", res["ULS3"]}]]];
If[OptionValue[Level] ≥ 4, res["ULS4"] =
  Optimizer[Prepend[constr, factor1 * FULS1 + FULS2 / Length[observed] +
     FULS3 / Length[observed]^2 + FULS4 / Length[observed]^3], val04];
 If[OptionValue[verbose], Print0[{"ULS4", res["ULS4"]}]]];
val0 = Map[{#, 0.5} &, GetAllVariables[{constr, Z2}]];
res["GLS"] = Optimizer[Prepend[constr,
   1 / 2 * Tr2[IdentityMatrix[Length[S2]] - Inverse[S2].Z2]], val0 ];
If[OptionValue[verbose], Print0[{"GLS", res["GLS"]}]];
If[OptionValue[doML],
 res["ML"] = Optimizer[Join[
    {Log[Abs[Det[Z2]]] + Tr[S2.Inverse[Z2]] - Log[Det[S2]] - Length[observed],
     Det[Z2] > 0}, constr], Map[{#[[1]], #[[2]]} &, res["ULS2"][[2]]]  ];
 If[OptionValue[verbose], Print0[{"ML", res["ML"]}]] ];
f = Module[{v = vect[S2] - vect[Z2]}, Dot[v, (WI.v)]];  res["f"] = f;
res["FWLS0"] = Optimizer[Prepend[constr, f], val0];
If[OptionValue[verbose], Print0[{"FWLS0", res["FWLS0"]}]];
res["FWLS"] =
 Optimizer[Prepend[constr, f], Map[{#[[1]], #[[2]]} &, res["ULS2"][[2]]]];
If[OptionValue[verbose], Print0[{"FWLS", res["FWLS"]}]];
If[Length[OptionValue[NDObs]] > 0,
 (* these vars are explicity assumed to be multivariate normal *)
 TakeByIndices[L_List, idx_List] := Table[L[[idx[[i]]]], {i, 1, Length[idx]}];
 MTakeByIndices[L_List, idx_List] :=
  Module[{M = Table[L[[idx[[i]]]], {i, 1, Length[idx]}]},
   Map[TakeByIndices[#, idx] &, M] ];
 IL = Table[Position[observed, OptionValue[NDObs][[i]]][[1, 1]],
   {i, 1, Length[OptionValue[NDObs]]}];
 S2a = MTakeByIndices[S2, IL]; Z2a = MTakeByIndices[Z2, IL];
 F1 = Log[Abs[Det[Z2a]]] + Tr[S2a.Inverse[Z2a]] -
   Log[Det[S2a]] - Length[OptionValue[NDObs]];
 S2b = Table[If[MemberQ[IL, i] && MemberQ[IL, j], 0, S2[[i, j]]],
   {i, 1, Length[S2]}, {j, 1, Length[S2]}];
 Z2b = Table[If[MemberQ[IL, i] && MemberQ[IL, j], 0, Z2[[i, j]]],
   {i, 1, Length[Z2]}, {j, 1, Length[Z2]}];
 F2 = 1 / 2 * Apply[Plus, Map[#^2 &, Flatten[S2b - Z2b]]];
 res["MMLFL"] = Optimizer[
   Join[{F1 + F2, Det[Z2a] > 0}, constr], Map[{#[[1]], #[[2]]} &, res["ULS2"][[2]]] ];
```

```
  If[OptionValue[verbose], Print0[{"MMLFL", res["MMLFL"]}]];
  res["FWLS2"] =
   Optimizer[Prepend[constr, f], Map[{#[[1]], #[[2]]} &, res["MMLFL"][[2]]]];
  If[OptionValue[verbose], Print0[{"FWLS2", res["FWLS2"]}]];
  res["F1"] = F1;
  res["F2"] = F2;
  res["Z2a"] = Z2a;
  res["Z2b"] = Z2b;
  f2 = Module[{v = vect[S2b] - vect[Z2b]}, Dot[v, (WI.v)]];
  res["MULFL"] = Optimizer[
    Join[{F1 + f2, Det[Z2a] > 0}, constr], Map[{#[[1]], #[[2]]} &, res["ULS2"][[2]] ]];
  If[OptionValue[verbose], Print0[{"MULFL", res["MULFL"]}]];
 ];
 res
]
```

In[ ]:=
```
(* Ganzach's
 model ********************************************************** *)
tp = {gamma1 → 0.3, gamma2 → 0.2, om11 → 0.5, om12 → 0.3,
   om22 → 0.4, Cov[xi1, xi2] → 0.2, Var[xi1] → 1, Var[xi2] → 1,
   Var[e1] → 0.3^2, Var[e2] → 0.2^2, Var[e3] → 0.1^2,
   Var[e4] → 0.3^2, Var[e5] → 0.2^2, Var[e6] → 0.3^2,
   Var[d1] → 0.1^2, Var[d2] → 0.2^2, Var[d3] → 0.1^2, Var[e0] → 0.2^2,
   lx12 → 0.7, lx13 → 1.2, lx52 → 0.5, lx62 → 0.9, ly12 → 0.8, ly13 → 1.3};
Gsimudat[n_] :=
  Module[{XI, XI1, XI2, Etas, xs1, xs2, xs3, xs4, xs5, xs6, ys1, ys2, ys3},
   XI = RandomVariate[MultinormalDistribution[{0, 0},
      {{Var[xi1], Cov[xi1, xi2]}, {Cov[xi1, xi2], Var[xi2]}} /. tp], n];
   XI1 = Transpose[XI][[1]]; XI2 = Transpose[XI][[2]];
   Etas = gamma1 * XI1 + gamma2 * XI2 + om11 * XI1^2 + om12 * XI1 * XI2 + om22 * XI2^2 +
      RandomVariate[NormalDistribution[0, Sqrt[Var[e0]] /. tp], n] /. tp;
xs1 = 1 * XI1 + RandomVariate[NormalDistribution[0, Sqrt[Var[e1]] /. tp], n];
   xs2 = (lx12 /. tp) * XI1 +
     RandomVariate[NormalDistribution[0, Sqrt[Var[e2]] /. tp], n];
xs3 = (lx13 /. tp) * XI1 + RandomVariate[
      NormalDistribution[0, Sqrt[Var[e3]] /. tp], n];
   xs4 = 1 * XI2 + RandomVariate[NormalDistribution[0, Sqrt[Var[e4]] /. tp], n];
xs5 = (lx52 /. tp) * XI2 +
     RandomVariate[NormalDistribution[0, Sqrt[Var[e5]] /. tp], n];
   xs6 = (lx62 /. tp) * XI2 + RandomVariate[
      NormalDistribution[0, Sqrt[Var[e6]] /. tp], n];
ys1 = 1 * Etas + RandomVariate[NormalDistribution[0, Sqrt[Var[d1]] /. tp], n];
   ys2 =
    (ly12 /. tp) * Etas + RandomVariate[NormalDistribution[0, Sqrt[Var[d2]] /. tp],
      n];

ys3 = (ly13 /. tp) * Etas +
     RandomVariate[NormalDistribution[0, Sqrt[Var[d3]] /. tp], n];
Transpose[{xs1, xs2, xs3, xs4, xs5, xs6, ys1, ys2, ys3}]];
```

```
In[•]:= Gsim = Gsimudat[25 000];
        Off[FindMinimum::precw];
        res = PolySem[Join[{{x1, x2, x3, x4, x5, x6, y1, y2, y3}}, Gsim], {
            y1 → eta + o1 + d1, y2 → ly12 * eta + o2 + d2, y3 → ly13 * eta + o3 + d3,
            x1 → xi1 + e1, x2 → lx12 * xi1 + e2, x3 → lx13 * xi1 + e3,
            x4 → xi2 + e4, x5 → lx52 * xi2 + e5, x6 → lx62 * xi2 + e6,
            eta → alpha + gamma1 * xi1 + gamma2 * xi2 +
              om11 * xi1^2 + om12 * xi1 * xi2 + om22 * xi2^2 + e0},
          {xi1, xi2}, {eta}, {e0, e1, e2, e3, e4, e5, e6, d1, d2, d3},
          {alpha, ly12, ly13, lx12, lx13, lx52, lx62, gamma1,
           gamma2, om11, om12, om22, o1, o2, o3}, {} , verbose → False,
          Level → 3, doML → False, NDObs → {x1, x2, x3, x4, x5, x6}];
        Print[Keys[res]];
        Do[Print[{m, Round[({gamma1, gamma2, om11, om12, om22} /. res[m][[2]]) -
              ({gamma1, gamma2, om11, om12, om22} /. tp), 0.00001]}],
         {m, {"ULS2", "ULS3", "GLS", "FWLS", "FWLS2", "MULFL", "MMLFL"}}]

Out[•]= {n, S1, Z1, S2, Z2, S3, Z3, S4, Z4, constr, ULS1, ULS2,
         ULS3, GLS, f, FWLS0, FWLS, MMLFL, FWLS2, F1, F2, Z2a, Z2b, MULFL}

        {ULS2, {-0.0072, 0.00864, -0.19418, -0.02475, -0.1226}}

        {ULS3, {-0.01214, 0.00504, -0.01233, 0.00022, -0.01593}}

        {GLS, {-0.00717, 0.00964, -0.30263, -0.11691, -0.26712}}

        {FWLS, {-0.00689, 0.01108, -0.19424, -0.02479, -0.12265}}

        {FWLS2, {-0.00689, 0.01108, -0.19642, -0.02368, -0.11956}}

        {MULFL, {-0.00647, 0.01106, -0.18516, -0.01128, -0.10571}}

        {MMLFL, {-0.00772, 0.00946, -0.19602, -0.0234, -0.11919}}

In[•]:=

In[•]:= res["Z2"][[9, 9]]

Out[•]= 2 gamma1 gamma2 ly13^2 Cov[xi1, xi2] + ly13^2 om12^2 Cov[xi1, xi2]^2 +
        4 ly13^2 om11 om22 Cov[xi1, xi2]^2 + Var[d3] + ly13^2 Var[e0] + gamma1^2 ly13^2 Var[xi1] +
        4 ly13^2 om11 om12 Cov[xi1, xi2] × Var[xi1] + 2 ly13^2 om11^2 Var[xi1]^2 +
        gamma2^2 ly13^2 Var[xi2] + 4 ly13^2 om12 om22 Cov[xi1, xi2] × Var[xi2] +
        ly13^2 om12^2 Var[xi1] × Var[xi2] + 2 ly13^2 om22^2 Var[xi2]^2
```

In[•]:=
```
meth = {"ULS2", "ULS3", "GLS", "FWLS", "FWLS2", "MULFL", "MMLFL"};
Do[Print[nn];
 results = Table[{}, Length[meth]];
 Do[Gsim = Gsimudat[nn];
  res = PolySem[Join[{{x1, x2, x3, x4, x5, x6, y1, y2, y3}}, Gsim], {
      y1 → eta + o1 + d1, y2 → ly12 * eta + o2 + d2, y3 → ly13 * eta + o3 + d3,
      x1 → xi1 + e1, x2 → lx12 * xi1 + e2, x3 → lx13 * xi1 + e3,
      x4 → xi2 + e4, x5 → lx52 * xi2 + e5, x6 → lx62 * xi2 + e6,
      eta → alpha + gamma1 * xi1 + gamma2 * xi2 +
        om11 * xi1^2 + om12 * xi1 * xi2 + om22 * xi2^2 + e0},
     {xi1, xi2}, {eta}, {e0, e1, e2, e3, e4, e5, e6, d1, d2, d3},
     {alpha, ly12, ly13, lx12, lx13, lx52, lx62, gamma1,
      gamma2, om11, om12, om22, o1, o2, o3}, {} , verbose → False,
     Level → 3, doML → False, NDObs → {x1, x2, x3, x4, x5, x6}];
  Do[AppendTo[results[[i]], ({gamma1, gamma2, om11, om12, om22} /. res[meth[[i]][[2]]) -
      ({gamma1, gamma2, om11, om12, om22} /. tp)], {i, 1, Length@meth}];
  If[k == 1, Do[Print[{m, Round[({gamma1, gamma2, om11, om12, om22} /. res[m][[2]]) -
        ({gamma1, gamma2, om11, om12, om22} /. tp), 0.00001]}], {m, meth}]],
  {k, 100}];
 Do[Print[{meth[[i]], Round[Map[Mean, Transpose[results[[i]]]], 0.001],
    Round[Map[StandardDeviation, Transpose[results[[i]]]], 0.001]}],
  {i, 1, Length@meth}], {nn, {1000, 10 000}}]
```

```
1000
{ULS2, {-0.02914, 0.03881, -0.20059, 0.02597, -0.11262}}
{ULS3, {0.01484, 0.08514, 0.03099, -0.0112, 0.06138}}
{GLS, {-0.0283, 0.0307, -0.29464, -0.10804, -0.22229}}
{FWLS, {-0.00349, 0.02328, -0.19535, 0.02991, -0.10779}}
{FWLS2, {-0.00349, 0.02328, -0.19279, 0.02104, -0.13221}}
{MULFL, {0.00009, 0.02719, -0.21771, -0.00524, -0.13139}}
{MMLFL, {-0.03013, 0.0353, -0.20293, 0.01353, -0.14113}}
{ULS2, {-0.004, -0.005, -0.212, -0.002, -0.118}, {0.062, 0.05, 0.022, 0.032, 0.022}}
{ULS3, {-0.006, -0.002, -0.014, -0.019, -0.006}, {0.087, 0.078, 0.051, 0.063, 0.051}}
{GLS, {-0.003, -0.004, -0.301, -0.114, -0.258}, {0.062, 0.05, 0.008, 0.011, 0.041}}
{FWLS, {-0.001, -0.001, -0.22, -0.008, -0.126}, {0.061, 0.051, 0.022, 0.03, 0.022}}
{FWLS2, {-0.001, -0.001, -0.221, -0.011, -0.127}, {0.061, 0.051, 0.035, 0.034, 0.039}}
{MULFL, {0.005, 0.004, -0.207, 0.008, -0.11}, {0.085, 0.077, 0.106, 0.179, 0.108}}
{MMLFL, {-0.004, -0.005, -0.212, -0.005, -0.118}, {0.062, 0.049, 0.034, 0.034, 0.038}}
10 000
{ULS2, {-0.01347, 0.00811, -0.19622, 0.00067, -0.12881}}
{ULS3, {-0.01904, 0.00839, -0.01658, -0.00378, 0.00836}}
{GLS, {-0.01398, 0.00867, -0.30028, -0.11522, -0.25386}}
{FWLS, {-0.01437, 0.00831, -0.19642, 0.00052, -0.12901}}
{FWLS2, {-0.01437, 0.00831, -0.19271, 0.00088, -0.1301}}
{MULFL, {-0.01308, 0.00908, -0.19692, -0.00195, -0.13449}}
{MMLFL, {-0.01242, 0.00807, -0.19164, 0.00165, -0.12913}}
{ULS2, {-0.001, 0., -0.2, -0.004, -0.119}, {0.021, 0.02, 0.012, 0.017, 0.01}}
{ULS3, {-0.002, -0.001, 0.001, -0.004, -0.002}, {0.032, 0.029, 0.02, 0.022, 0.018}}
{GLS, {-0.001, 0., -0.301, -0.116, -0.257}, {0.021, 0.02, 0.002, 0.003, 0.011}}
{FWLS, {0., 0., -0.201, -0.005, -0.12}, {0.021, 0.02, 0.012, 0.017, 0.01}}
{FWLS2, {0., 0., -0.199, -0.003, -0.118}, {0.021, 0.02, 0.016, 0.023, 0.019}}
{MULFL, {0., 0., -0.199, -0.003, -0.118}, {0.021, 0.02, 0.017, 0.02, 0.016}}
{MMLFL, {-0.001, 0., -0.198, -0.002, -0.116}, {0.021, 0.019, 0.016, 0.023, 0.019}}
```

In[•]:= **(********** Muthen's model **********)**

In[•]:=
```
meth = {"ULS2", "ULS3", "GLS", "FWLS"};
Do[Print[nn];
 results = Table[{}, Length[meth]];
 Do[
Etas =
    RandomVariate[MultinormalDistribution[{0, 0}, {{1.2, 0.4}, {0.4, 0.8}}], nn];
  Eta1 = Transpose[Etas][[1]]; Eta2 = Transpose[Etas][[2]];
B1 = 0.1; B2 = 0.3; B3 = 0.2;  B4 = .7;
Eta3 = B1 * Eta1 + B2 * Eta2 + B3 * Eta1 * Eta2 +
    RandomVariate[NormalDistribution[0, 0.2], nn];
Eta4 = B4 * Eta3 + RandomVariate[NormalDistribution[0, 0.1], nn];
ys1 = 1 * Eta1 + RandomVariate[NormalDistribution[0, 0.1], nn];
  ys2 = 0.5 * Eta1 + RandomVariate[NormalDistribution[0, 0.2], nn];
  ys3 = 0.7 * Eta1 + RandomVariate[NormalDistribution[0, 0.3], nn];
ys4 = 1 * Eta2 + RandomVariate[NormalDistribution[0, 0.1], nn];
  ys5 = 0.7 * Eta2 + RandomVariate[NormalDistribution[0, 0.2], nn];
  ys6 = 0.4 * Eta2 + RandomVariate[NormalDistribution[0, 0.3], nn];
ys7 = 1 * Eta3 + RandomVariate[NormalDistribution[0, 0.1], nn];
  ys8 = 1.2 * Eta3 + RandomVariate[NormalDistribution[0, 0.2], nn];
  ys9 = 0.4 * Eta3 + RandomVariate[NormalDistribution[0, 0.3], nn];
ys10 = 1 * Eta4 + RandomVariate[NormalDistribution[0, 0.1], nn];
  ys11 = 0.8 * Eta4 + RandomVariate[NormalDistribution[0, 0.2], nn];
  ys12 = 0.9 * Eta4 + RandomVariate[NormalDistribution[0, 0.3], nn];
Msimudat =
    Transpose[{ys1, ys2, ys3, ys4, ys5, ys6, ys7, ys8, ys9, ys10, ys11, ys12}];
  res = PolySem[Join[{{y1, y2, y3, y4, y5, y6, y7, y8, y9, y10, y11, y12}},
     Msimudat], {
     y1 → 1 * eta1 + oy1 + e1, y2 → c2 * eta1 + oy2 + e2, y3 → c3 * eta1 + oy3 + e3,
     y4 → 1 * eta2 + oy4 + e4, y5 → c5 * eta2 + oy5 + e5, y6 → c6 * eta2 + oy6 + e6,
     y7 → 1 * eta3 + oy7 + e7, y8 → c8 * eta3 + oy8 + e8, y9 → c9 * eta3 + oy9 + e9, y10 →
      1 * eta4 + oy10 + e10, y11 → c11 * eta4 + oy11 + e11, y12 → c12 * eta4 + oy12 + e12,
     eta3 → b1 * eta1 + b2 * eta2 + b3 * eta1 * eta2 + o3 + d1, eta4 → b4 * eta3 + o4 + d2},
    {eta1, eta2}, {eta3, eta4},  {e1, e2, e3, e4, e5, e6,
     e7, e8, e9, e10, e11, e12, d1, d2},
    {c2, c3, c5, c6, c8, c9, c11, c12, b1, b2, b3, b4, oy1, oy2, oy3, oy4, oy5,
     oy6, oy7, oy8, oy9, oy10, oy11, oy12, o4, o3}, {}, Level → 3, verbose → False];
  If[k < 0, Print[res["ULS2"]]; Print[res["ULS3"]]];
  Do[
   AppendTo[results[[i]], ({b1, b2, b3, b4} /. res[meth[[i]]][[2]]) - {B1, B2, B3, B4}],
   {i, 1, Length@meth}], {k, 10}];
 Do[Print[{meth[[i]], Round[Map[Mean, Transpose[results[[i]]]], 0.001],
    Round[Map[StandardDeviation, Transpose[results[[i]]]], 0.001]}],
  {i, 1, Length@meth}], {nn, {1000, 10 000}}]
```

```
1000
{ULS2, {-0.006, -0.008, -0.206, -0.005}, {0.007, 0.019, 0.008, 0.015}}
{ULS3, {-0.006, -0.006, 0.005, -0.007}, {0.007, 0.02, 0.015, 0.012}}
{GLS, {-0.005, -0.007, 0.114, -0.008}, {0.008, 0.019, 0.004, 0.011}}
{FWLS, {-0.003, -0.005, -0.206, -0.008}, {0.008, 0.019, 0.008, 0.012}}
10 000
```

In[•]:=
```
Timing[Gsim = Gsimudat[1000];
 res = PolySem[Join[{{x1, x2, x3, x4, x5, x6, y1, y2, y3}}, Gsim], {
    y1 → eta + o1 + d1, y2 → ly12 * eta + o2 + d2, y3 → ly13 * eta + o3 + d3,
    x1 → xi1 + e1, x2 → lx12 * xi1 + e2, x3 → lx13 * xi1 + e3,
    x4 → xi2 + e4, x5 → lx52 * xi2 + e5, x6 → lx62 * xi2 + e6,
    eta → alpha + gamma1 * xi1 + gamma2 * xi2 +
      om11 * xi1^2 + om12 * xi1 * xi2 + om22 * xi2^2 + e0},
   {xi1, xi2}, {eta}, {e0, e1, e2, e3, e4, e5, e6, d1, d2, d3},
   {alpha, ly12, ly13, lx12, lx13, lx52, lx62, gamma1,
    gamma2, om11, om12, om22, o1, o2, o3}, {} , verbose → False,
   Level → 3, doML → False, NDObs → {x1, x2, x3, x4, x5, x6}];]
```

••• FindMinimum: The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances.

••• FindMinimum: The line search decreased the step size to within the tolerance specified by AccuracyGoal and PrecisionGoal but was unable to find a sufficient decrease in the function. You may need more than MachinePrecision digits of working precision to meet these tolerances.

Out[•]=
```
{193.269, Null}
```