Title: Tutorial 4B: Tour through the Hyperion Library

Speakers: Aike Liu

Collection: Mini-Course of Numerical Conformal Bootstrap

Date: April 27, 2023 - 3:30 PM

URL: https://pirsa.org/23040147

Hyperion Tutorial 3

Advanced Search Methods

Bootstrap Mini Course: Tutorial 4b

Aike Liu

California Institute of Technology

Caltech PERIMETER INSTITUTE

SIMONS FOUNDATION

`scalars-3d/fermions-3d`

`Bound.hs`

Crossing equations

OPE coefficients

Set up SDP

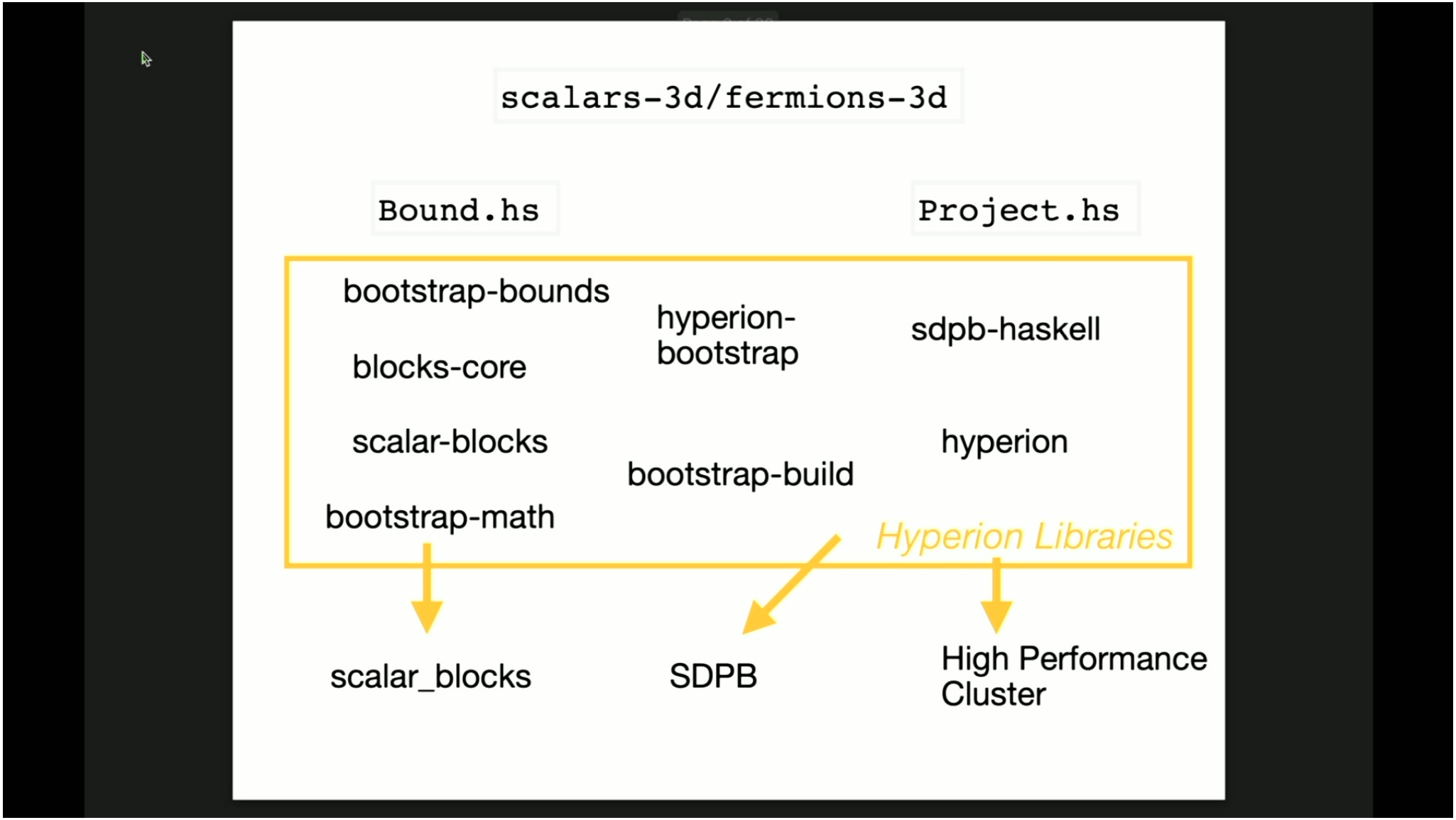
`Project.hs`

Impose gaps, set up OPE search

Numerical parameters

such as Λ, κ (pole order), precisions and SDPB parameters

submit HPC remote job


scalars-3d/fermions-3d
Bound.hs
Project.hs
bootstrap-bounds
blocks-core
scalar-blocks
bootstrap-math
hyperion-bootstrap
bootstrap-build
sdpb-haskell
hyperion
Hyperion Libraries
scalar_blocks
SDPB
High Performance Cluster

`Project/Fermions3d/GNYSigPsiEpsTest2020.hs`

1. Decide your spectrum and other physical input.
2. Decide your numerical details.
3. Decide algorithms to use, e.g., feasibility tests, Delaunay search, Cutting surface …
4. Set up remote computations.

```haskell
boundsProgram :: Text -> Cluster ()

boundsProgram "GNYSigPsiEps_test_ngroup2_nmax6" =
  local (setJobType (MPIJob 2 24) . setJobTime (8*hour) . setSlurmPartition "compute") $ voi
  mapConcurrently_ (Bound.remoteCompute . bound) $
      fmap toV $ [(1.0686, 0.651, 1.70)] --n2nmax6PrimalPoints --GNYData.epsN2nmax6DualPoints
  where
    nmax = 6
    ngroup = 2
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts Nothing ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , dualErrorThreshold=1e-30
                                                       , primalErrorThreshold=1e-40
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

MPIJob #Nodes #Tasks per node

Concurrently type represents a computation that can be run concurrently with other Concurrently-type

```haskell
mapConcurrently_ :: (Applicative (Concurrently m), Foldable t, Functor t) => (a -> m b) -> t a -> m ()
mapConcurrently_ f = doConcurrently_ . fmap f
```

```
boundsProgram "GNYSigPsiEps_test_ngroup2_nmax6" =
  local (setJobType (MPIJob 2 24) . setJobTime (8*hour) . setSlurmPartition "compute") $ voi
  mapConcurrently_ (Bound.remoteCompute . bound) $
     fmap toV $ [(1.0686, 0.651, 1.70)] --n2nmax6PrimalPoints --GNYData.epsN2nmax6DualPoints
  where
    nmax = 6
    ngroup = 2
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts Nothing ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , dualErrorThreshold=1e-30
                                                       , primalErrorThreshold=1e-40
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

Call sdp2input and sdpb

```
remoteCompute
  :: ( Typeable b
     , Static (Show b)
     , Static (ToJSON b)
     , Static (ToSDP b)
     , Static (Binary b)
     , Static (SDPFetchBuildConfig b)
     , Static (BuildInJob b)
     )
  => Bound Int b
  -> Cluster SDPB.Output
remoteCompute =
  remoteComputeWithFileTreatment keepOutAnd
```

```haskell
boundsProgram :: Text -> Cluster ()

boundsProgram "IsingSigEpsAllowed_test_nmax6" =
  local (setJobType (MPIJob 1 6) . setJobTime (45*minute)) $
  mapConcurrently_ (Bound.remoteCompute . bound)
  [ (toV (0.517075, 1.396575), thetaVectorApprox 1e-16 0.964)
  , (toV (0.518149, 1.412625), thetaVectorApprox 1e-16 0.96926)
  ]
  where
    nmax = 6
    bound (deltaExts, lambda) = Bound
      { boundKey = isingFeasibleDefaultGaps deltaExts lambda nmax
      , precision = SB.precision (Params.blockParamsNmax nmax :: SB.ScalarBlockPa
      , solverParams = (Params.sdpbParamsNmax nmax) { precision = 768 }
      , boundConfig = defaultBoundConfig
      }
```

```haskell
data Bound prec a = Bound
  { boundKey     :: a
  , precision    :: prec
  , solverParams :: SDPB.Params
  , boundConfig  :: BoundConfig
  } deriving (Show, Eq, Ord, Generic, Binary, ToJSON, FromJSON, Functor)
```

```
boundsProgram "GNYSigPsiEps_OPEtest_ngroup2_nmax6" =
  local (setJobType (MPIJob 1 18) . setJobTime (8*hour)) $ void $ do
  mapConcurrently_ (Bound.remoteCompute . bound) $
     fmap toV GNYData.epsN2nmax6PrimalPoints --n2nmax6PrimalPoints --GNYData.epsN2n
  where
    nmax = 6
    ngroup = 2
    mLambda :: Maybe (V 4 Rational)
    mLambda = Just $ toV (0,1,1,1)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts mLambda ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

```haskell
boundsProgram :: Text -> Cluster ()

boundsProgram "IsingSigEpsAllowed_test_nmax6" =
  local (setJobType (MPIJob 1 6) . setJobTime (45*minute)) $
  mapConcurrently_ (Bound.remoteCompute . bound)
  [ (toV (0.517075, 1.396575), thetaVectorApprox 1e-16 0.964)
  , (toV (0.518149, 1.412625), thetaVectorApprox 1e-16 0.96926)
  ]
  where
    nmax = 6
    bound (deltaExts, lambda) = Bound
      { boundKey = isingFeasibleDefaultGaps deltaExts lambda nmax
      , precision = SB.precision (Params.blockParamsNmax nmax :: SB.ScalarBlockPa
      , solverParams = (Params.sdpbParamsNmax nmax) { precision = 768 }
      , boundConfig = defaultBoundConfig
      }
```

```haskell
data Bound prec a = Bound
  { boundKey     :: a
  , precision    :: prec
  , solverParams :: SDPB.Params
  , boundConfig  :: BoundConfig
  } deriving (Show, Eq, Ord, Generic, Binary, ToJSON, FromJSON, Functor)
```

```
boundsProgram "GNYSigPsiEps_OPEtest_ngroup2_nmax6" =
  local (setJobType (MPIJob 1 18) . setJobTime (8*hour)) $ void $ do
  mapConcurrently_ (Bound.remoteCompute . bound) $
     fmap toV GNYData.epsN2nmax6PrimalPoints --n2nmax6PrimalPoints --GNYData.epsN2n
  where
    nmax = 6
    ngroup = 2
    mLambda :: Maybe (V 4 Rational)
    mLambda = Just $ toV (0,1,1,1)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts mLambda ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

```haskell
gnySigPsiEpsFeasibleDefaultGaps ::
       V 3 Rational -> Maybe (V 4 Rational) -> Rational -> Int -> GNYSigPsiEps.GNYSigPsiEps

gnySigPsiEpsFeasibleDefaultGaps deltaExts mLambda ngroup nmax = GNYSigPsiEps.GNYSigPsiEps
  { spectrum     = Spectrum.setTwistGap 1e-6 $ Spectrum.setGaps
    [ (psiChannel, 2)
    , (chiChannel, 3.5) --3.5)
    , (sigChannel, 2.5) --2.5)
    , (epsChannel, 3) --3)
    , (sigTChannel, 2)
      -- , (epsTChannel, 3)
    ] Spectrum.unitarySpectrum
  , nGroup       = ngroup
  , objective    = GNYSigPsiEps.Feasibility mLambda
  , externalDims = GNYSigPsiEps.ExternalDims {..}
  , blockParams  = Params.block3dParamsNmax nmax
  , spins        = Params.gnyspinsNmax nmax
  }
  where
    (deltaPsi, deltaSig, deltaEps) = fromV deltaExts
```

```
boundsProgram "GNYSigPsiEps_test_ngroup2_nmax6" =
  local (setJobType (MPIJob 2 24) . setJobTime (8*hour) . setSlurmPartition "compute")
  mapConcurrently_ (Bound.remoteCompute . bound) $
     fmap toV $ [(1.0686, 0.651, 1.70)] --n2nmax6PrimalPoints --GNYData.epsN2nmax6Dual
  where
    nmax = 6
    ngroup = 2
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts Nothing ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , dualErrorThreshold=1e-30
                                                       , primalErrorThreshold=1e-40
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

```
data Bound prec a = Bound
  { boundKey     :: a
  , precision    :: prec
  , solverParams :: SDPB.Params
  , boundConfig  :: BoundConfig
  } deriving (Show, Eq, Ord, Generic, Binary, ToJSON, FromJSON, Functor)
```

```
boundsProgram "GNYSigPsiEps_OPEtest_ngroup2_nmax6" =
  local (setJobType (MPIJob 1 18) . setJobTime (8*hour)) $ void $ do
  mapConcurrently_ (Bound.remoteCompute . bound) $
     fmap toV GNYData.epsN2nmax6PrimalPoints --n2nmax6PrimalPoints --GNYData.epsN2n
  where
    nmax = 6
    ngroup = 2
    mLambda :: Maybe (V 4 Rational)
    mLambda = Just $ toV (0,1,1,1)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts mLambda ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

```
boundsProgram "GNYSigPsiEps_OPEtest_ngroup2_nmax6" =
  local (setJobType (MPIJob 1 18) . setJobTime (8*hour)) $ void $ do
  mapConcurrently_ (Bound.remoteCompute . bound) $
     fmap toV GNYData.epsN2nmax6PrimalPoints --n2nmax6PrimalPoints --GNYData.epsN2nm
  where
    nmax = 6
    ngroup = 2
    mLambda :: Maybe (V 4 Rational)
    mLambda = Just $ toV (0,1,1,1)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts mLambda ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

```
jumpFindingParams :: Int -> SDPB.Params
jumpFindingParams n = (sdpbParamsNmax n)
  { SDPB.dualErrorThreshold       = 1e-200
  , SDPB.primalErrorThreshold     = 1e-200
  , SDPB.findDualFeasible         = False
  , SDPB.findPrimalFeasible       = False
  , SDPB.detectPrimalFeasibleJump = True
  , SDPB.detectDualFeasibleJump   = True
  }
```

Delaunay

```
boundsProgram "GNYSigPsiEps_island_ngroup2_nmax6" =
 local (setJobType (MPIJob 1 16) . setJobTime (8*hour)) $ do
  void $ delaunaySearchRegionPersistent delaunaySearchPoints delaunayConfig affine initialPts f
  where
    nmax = 6
    ngroup = 2
    affine = GNYData.gny2Affine3dWindow
    f :: V 3 Rational -> Cluster Bool
    f deltaExts = fmap SDPB.isPrimalFeasible (Bound.remoteCompute $ bound deltaExts)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts Nothing ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , dualErrorThreshold=1e-30
                                                       , primalErrorThreshold=1e-40
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
    delaunayConfig = defaultDelaunayConfig 20 200
    initialPts = Map.fromList $
      [(p, Just True)  | p <- initialAllowed] ++ [(p, Just False) | p <- initialDisallowed]
      where
        initialDisallowed = fmap toV $ GNYData.epsN2nmax6DualPoints
        initialAllowed = fmap toV $ GNYData.epsN2nmax6PrimalPoints
```

Delaunay

```
boundsProgram "GNYSigPsiEps_island_ngroup2_nmax6" =
 local (setJobType (MPIJob 1 16) . setJobTime (8*hour)) $ do
  void $ delaunaySearchRegionPersistent delaunaySearchPoints delaunayConfig affine initialPts f
```

```
delaunaySearchPoints :: DB.KeyValMap (V n Rational) (Maybe Bool)
delaunaySearchPoints = DB.KeyValMap "delaunaySearchPoints"
```

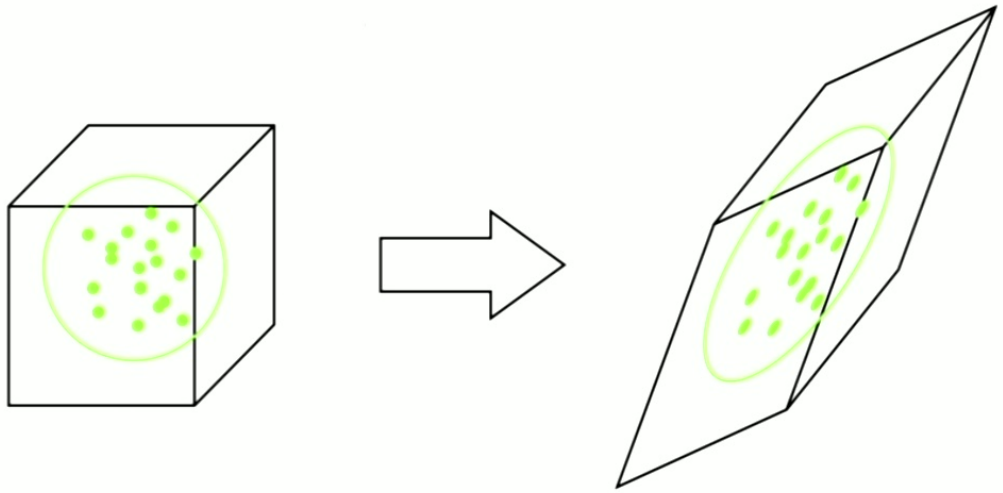
Delaunay

```
boundsProgram "GNYSigPsiEps_island_ngroup2_nmax6" =
 local (setJobType (MPIJob 1 16) . setJobTime (8*hour)) $ do
  void $ delaunaySearchRegionPersistent delaunaySearchPoints delaunayConfig affine initialPts f
  where
    nmax = 6
    ngroup = 2
    affine = GNYData.gny2Affine3dWindow
    f :: V 3 Rational -> Cluster Bool
    f deltaExts = fmap SDPB.isPrimalFeasible (Bound.remoteCompute $ bound deltaExts)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts Nothing ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , dualErrorThreshold=1e-30
                                                       , primalErrorThreshold=1e-40
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
    delaunayConfig = defaultDelaunayConfig 20 200
    initialPts = Map.fromList $
      [(p, Just True)  | p <- initialAllowed] ++ [(p, Just False) | p <- initialDisallowed]
      where
        initialDisallowed = fmap toV $ GNYData.epsN2nmax6DualPoints
        initialAllowed = fmap toV $ GNYData.epsN2nmax6PrimalPoints
```

20 threads, test 20 points each iteration.
200 number of points in total.

Delaunay

```
boundsProgram "GNYSigPsiEps_island_ngroup2_nmax6" =
 local (setJobType (MPIJob 1 16) . setJobTime (8*hour)) $ do
  void $ delaunaySearchRegionPersistent delaunaySearchPoints delaunayConfig affine initialPts f
  where
    nmax = 6
    ngroup = 2
    affine = GNYData.gny2Affine3dWindow
    f :: V 3 Rational -> Cluster Bool
    f deltaExts = fmap SDPB.isPrimalFeasible (Bound.remoteCompute $ bound deltaExts)
```



Delaunay

```
boundsProgram "GNYSigPsiEps_island_ngroup2_nmax6" =
 local (setJobType (MPIJob 1 16) . setJobTime (8*hour)) $ do
  void $ delaunaySearchRegionPersistent delaunaySearchPoints delaunayConfig affine initialPts f
  where
    nmax = 6
    ngroup = 2
    affine = GNYData.gny2Affine3dWindow
    f :: V 3 Rational -> Cluster Bool
    f deltaExts = fmap SDPB.isPrimalFeasible (Bound.remoteCompute $ bound deltaExts)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts Nothing ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , dualErrorThreshold=1e-30
                                                       , primalErrorThreshold=1e-40
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
    delaunayConfig = defaultDelaunayConfig 20 200
    initialPts = Map.fromList $
      [(p, Just True)  | p <- initialAllowed] ++ [(p, Just False) | p <- initialDisallowed]
      where
        initialDisallowed = fmap toV $ GNYData.epsN2nmax6DualPoints
        initialAllowed = fmap toV $ GNYData.epsN2nmax6PrimalPoints
```

20 threads, test 20 points each iteration.
200 number of points in total.

Delaunay

```
boundsProgram "GNYSigPsiEps_island_ngroup2_nmax6" =
 local (setJobType (MPIJob 1 16) . setJobTime (8*hour)) $ do
  void $ delaunaySearchRegionPersistent delaunaySearchPoints delaunayConfig affine initialPts f
  where
    nmax = 6
    ngroup = 2
    affine = GNYData.gny2Affine3dWindow
    f :: V 3 Rational -> Cluster Bool
    f deltaExts = fmap SDPB.isPrimalFeasible (Bound.remoteCompute $ bound deltaExts)
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts Nothing ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768
                                                       , dualErrorThreshold=1e-30
                                                       , primalErrorThreshold=1e-40
                                                       , findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
    delaunayConfig = defaultDelaunayConfig 20 200
    initialPts = Map.fromList $
      [(p, Just True)  | p <- initialAllowed] ++ [(p, Just False) | p <- initialDisallowed]
      where
        initialDisallowed = fmap toV $ GNYData.epsN2nmax6DualPoints
        initialAllowed = fmap toV $ GNYData.epsN2nmax6PrimalPoints
```

OPE Search

```
boundsProgram "GNYSigPsiEps_OPESearchTest_ngroup2_nmax6" =
  local (setJobType (MPIJob 1 18) . setJobTime (8*hour)) $ void $ do
  checkpointMap <- Bound.newCheckpointMap affine deltaExtToV Nothing
  lambdaMap     <- Bound.newLambdaMap affine deltaExtToV Nothing
  mapConcurrently_ (remoteGNYSigPsiEpsOPESearch checkpointMap lambdaMap initialBilinearForms . bound) $
     fmap toV [(1.0686, 0.651, 1.70)] --n2nmax6PrimalPoints --GNYData.epsN2nmax6DualPoints --[(1.06 + x/
  where
    nmax = 6
    ngroup = 2
    affine = GNYData.gny2Affine3dWindow--gny2Affine3dNmax6
    bilinearMat :: Matrix 4 4 Rational
    bilinearMat = toM
     ( (2226.2016, -2181.1580, -693.6937, 53.1024)
     , (-2181.1580, 4852.3925, -20.5093, -28.4479)
     , (-693.6937, - 20.5093, 413.5043, -33.8579)
     , (53.1024, -28.4479, -33.8579, 7.4766)
     )
    initialBilinearForms = BilinearForms 1e-32 [(Nothing, bilinearMat)]
    mLambda :: Maybe (V 4 Rational)
    mLambda = Just $ toV (0.3501, 0.165, 0.6787, 1) --(-0.258492, -0.123751, -0.515709,
    bound deltaExts = Bound
      { boundKey = gnySigPsiEpsFeasibleDefaultGaps deltaExts mLambda ngroup nmax
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.jumpFindingParams nmax) { precision = 768 }
      , boundConfig = defaultBoundConfig
      }
```

Delaunay + OPE Search

```
boundsProgram "GNYSigPsiEps_delaunay_OPEScan_ngroup2_nmax6" =
  gnyDelaunaySearchOPEScan ngroup searchData
  where
    ngroup = 2
--    bilinearMat :: Matrix 4 4 Rational
--    bilinearMat = toM
--     ( (2226.2016, -2181.1580, -693.6937, 53.1024)
--     , (-2181.1580, 4852.3925, -20.5093, -28.4479)
--     , (-693.6937, - 20.5093, 413.5043, -33.8579)
--     , (53.1024, -28.4479, -33.8579, 7.4766)
--     )
    lambda :: V 4 Rational
    lambda = toV (-0.258492, -0.123751, -0.515709, -0.807397)
    searchData = GNYDelaunaySearchData
        { nmax = 6
        , affine = GNYData.gny2OpeSig25AffineNmax6
        , initialLambda = lambda
        , initialCheckpoint = Nothing
        , initialDisallowed = fmap toV $ GNYData.opeSig25N2nmax6DualPoints
        , initialAllowed = fmap toV $ GNYData.opeSig25N2nmax6PrimalPoints
        , opeEllipse = GNYData.gny2Sig25bilinearMatNmax6
        , solverPrecision = 768
        , delaunayConfig = defaultDelaunayConfig 15 300
```

Delaunay + OPE Search

```
boundsProgram "GNYSigPsiEps_delaunay_OPEScan_ngroup2_nmax6" =
  gnyDelaunaySearchOPEScan ngroup searchData
```

```
gnyDelaunaySearchOPEScan :: Rational -> GNYDelaunaySearchData -> Cluster ()
gnyDelaunaySearchOPEScan nGroup GNYDelaunaySearchData{..} =
  local (setJobType jobType . setJobTime jobTime . setJobMemory jobMemory . setSlurmParti
  checkpointMap <- Bound.newCheckpointMap affine deltaExtToV initialCheckpoint
  lambdaMap     <- Bound.newLambdaMap affine deltaExtToV (Just initialLambda)
  delaunaySearchRegionPersistent delaunaySearchPoints delaunayConfig affine initialPts
    (remoteGNYSigPsiEpsOPESearch checkpointMap lambdaMap
     (BilinearForms 1e-32 [(Nothing, opeEllipse)]) . bound)
```

```haskell
boundsProgram "GNYSigPsiEps_GFF_test_ngroup2_nmax6" =
  local (setJobType (MPIJob 1 32) . setJobTime (24*hour)) $ void $
  mapConcurrently_ (remoteComputeGNYSigPsiEpsBounds . map bound) $ chunksOf 50 $ do
  (deltaPsi, deltaSig, deltaEps) <- [(1.06644, 0.657704, 1.75543)]
  --deltaPsi <- [1.055, 1.060 .. 1.075]
  --deltaSig <- [0.69, 0.71 .. 0.75]-- .. 0.651]
  --deltaEps <- [1.75]-- .. 1.725]
  pure (deltaPsi, deltaSig, deltaEps)
  where
    nmax = 6
    ngroup = 2
    --deltaPsi = 1.0686
    mLambda :: Maybe (V 4 Rational)
    mLambda = Just $ toV (-0.254569, -0.11829, -0.501378, -0.818389)
    bound (deltaPsi, deltaSig, deltaEps)  = Bound
      { boundKey = gnySigPsiEpsGFFNavigatorDefaultGaps (toV (deltaPsi, deltaSig, delta
      , precision = B3d.precision (Params.block3dParamsNmax nmax)
      , solverParams = (Params.optimizationParams nmax) { precision = 768}
                                                --, dualErrorThreshold=1e-30
                                                --, primalErrorThreshold=1e-40}
                                                --, findPrimalFeasible = False }
      , boundConfig = defaultBoundConfig
      }
```

```
boundsProgram "GNYSigPsiEps_GFFNavigatorBFGS_ngroup2_nmax6" =
  local (setJobType (MPIJob 1 32) . setJobTime (48*hour) . setJobMemory "90G" . setSlurmPartition "shared") $
  do
    result :: (V 3 Rational, [BFGS.BFGSData n (Rounded 'TowardZero 200)]) <-
      SDPDeriv.remoteBFGSBound bfgsConfig bjConfig
    Log.info "Finished BFGS search" result
```

```
$ ./hyp GNYSigPsiEps_island_ngroup2_nmax6
$ nohup: appending output to 'nohup.out'
$ vim nohup.out
```

```
[Fri 04/14/23 16:55:41] Program id: ProgramId "kWLzF"
[Fri 04/14/23 16:55:41] Process id: 170037
[Fri 04/14/23 16:55:41] Program arguments: BoundsProgramOptions
                    { programName = "GNYSigPsiEps_island_ngroup2_nmax6"
                    , initialDatabase = Nothing
                    }
[Fri 04/14/23 16:55:41] Using database: "/central/groups/dssimmon/aike/test/databases/
2023-04/kWLzF.sqlite"
[Fri 04/14/23 16:55:41] Running hold server on port: 44833
[Fri 04/14/23 16:55:41] Logging to: "/central/groups/dssimmon/aike/test/logs/2023-04/
kWLzF.log"
```

```
/central/groups/dssimmon/aike/test/logs/2023-04/kWLzF.log
/central/groups/dssimmon/aike/test/logs/2023-04/kWLzF/0.log

/central/groups/dssimmon/aike/test/databases/2023-04/kWLzF.sqlite
```

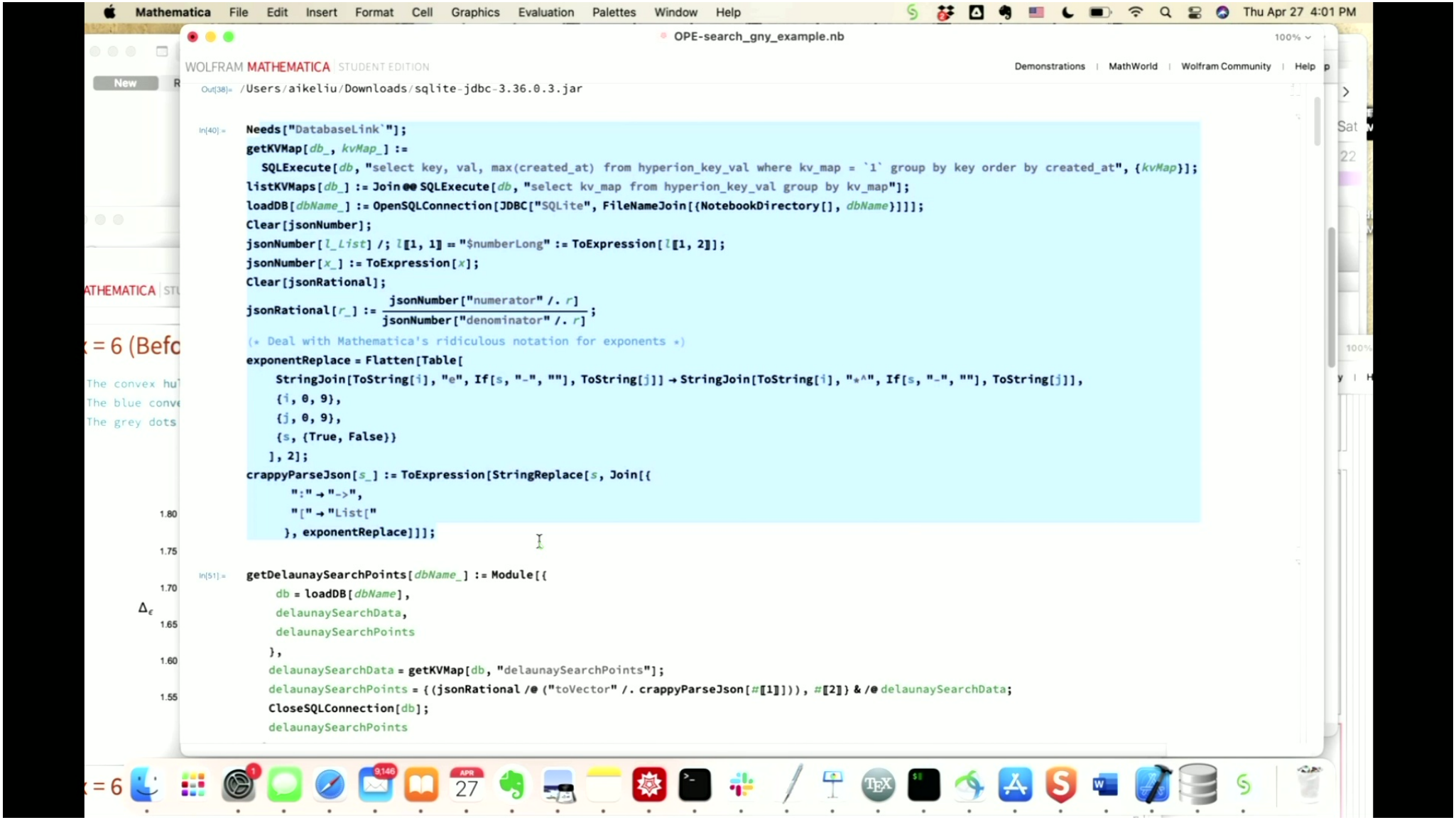
Setup

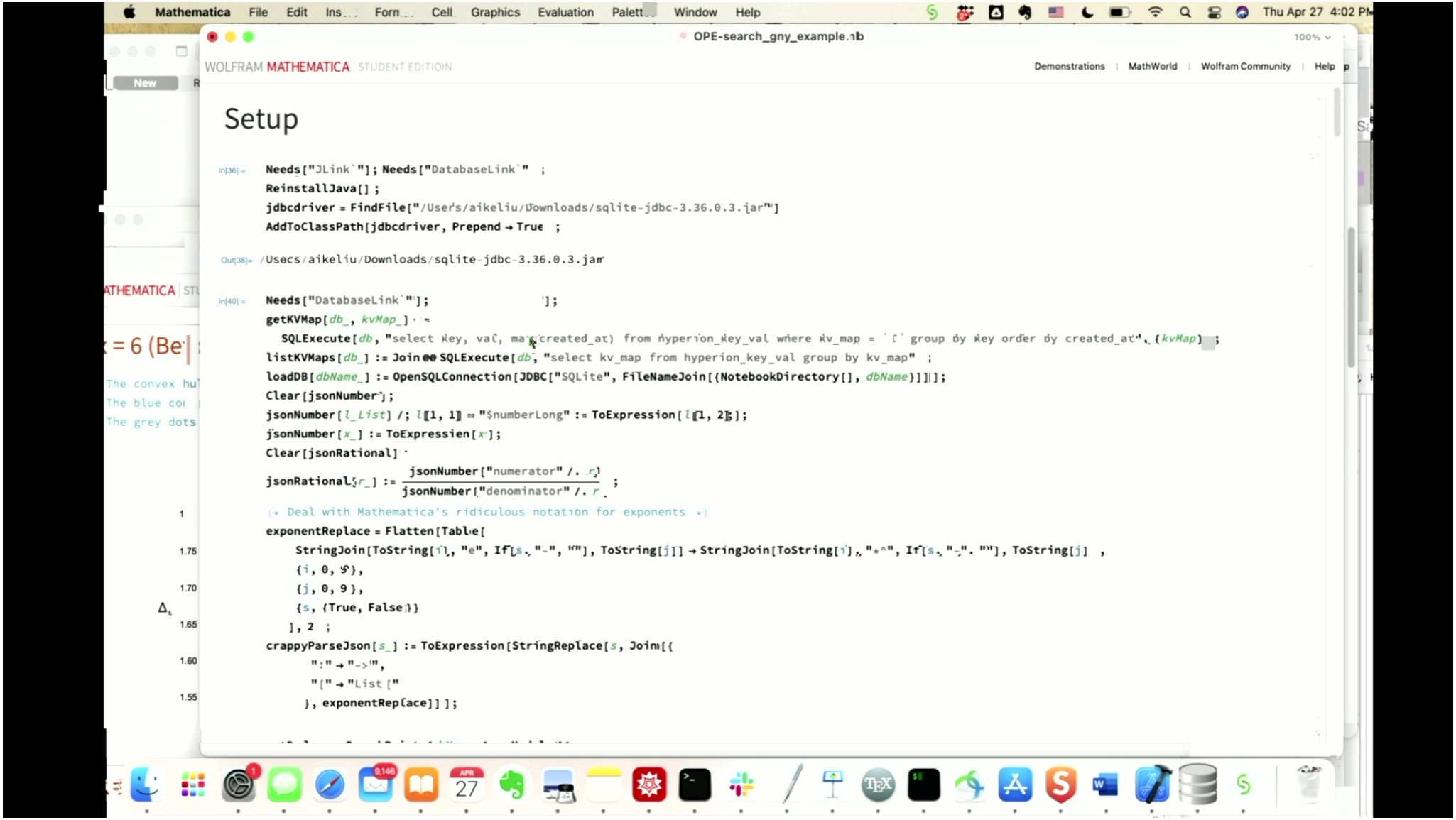
In[36]:=
```
Needs["JLink`"]; Needs["DatabaseLink`"];
ReinstallJava[];
jdbcdriver = FindFile["/Users/aikeliu/Downloads/sqlite-jdbc-3.36.0.3.jar"]
AddToClassPath[jdbcdriver, Prepend → True];
```

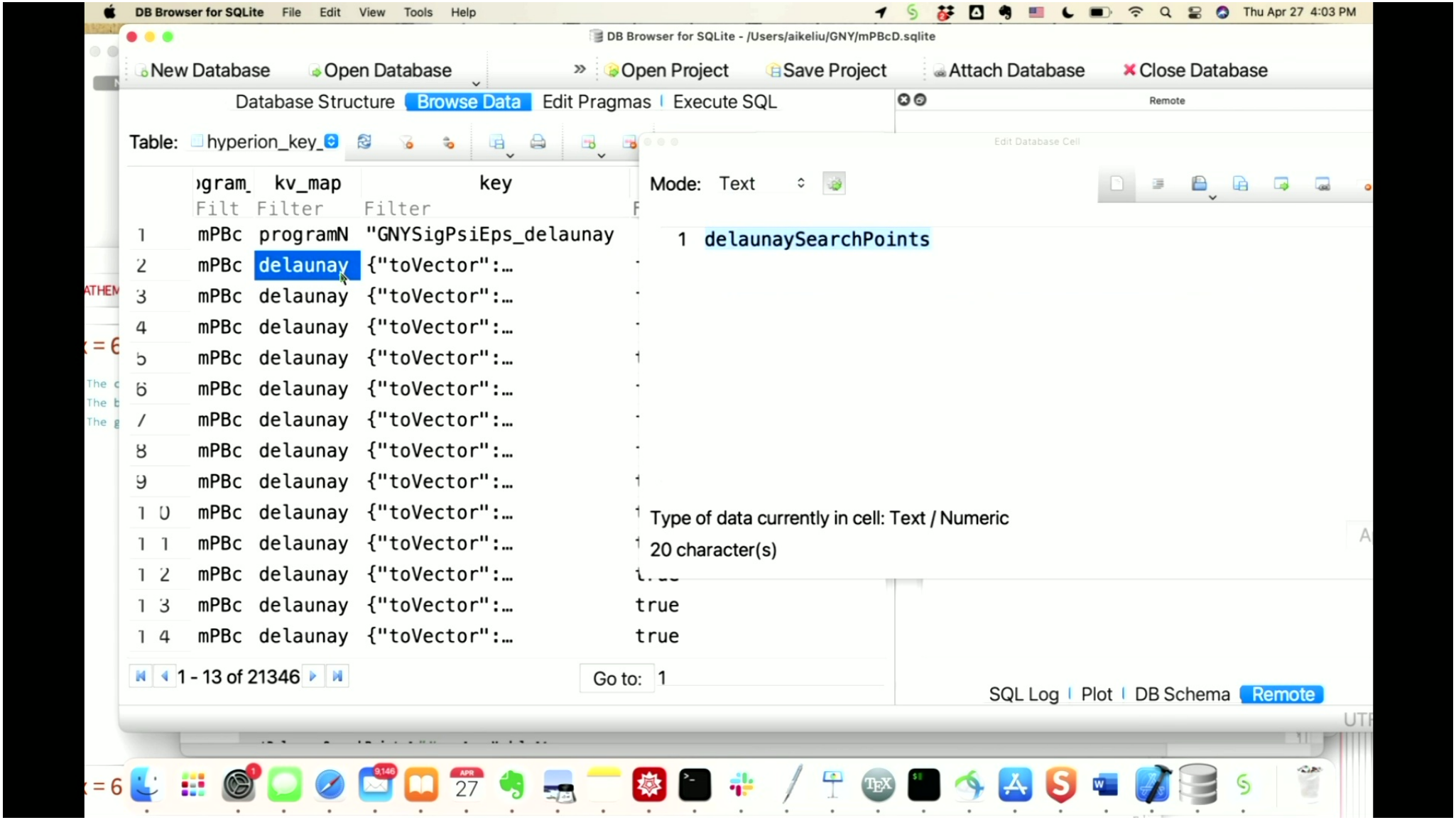
Out[38]= /Users/aikeliu/Downloads/sqlite-jdbc-3.36.0.3.jar

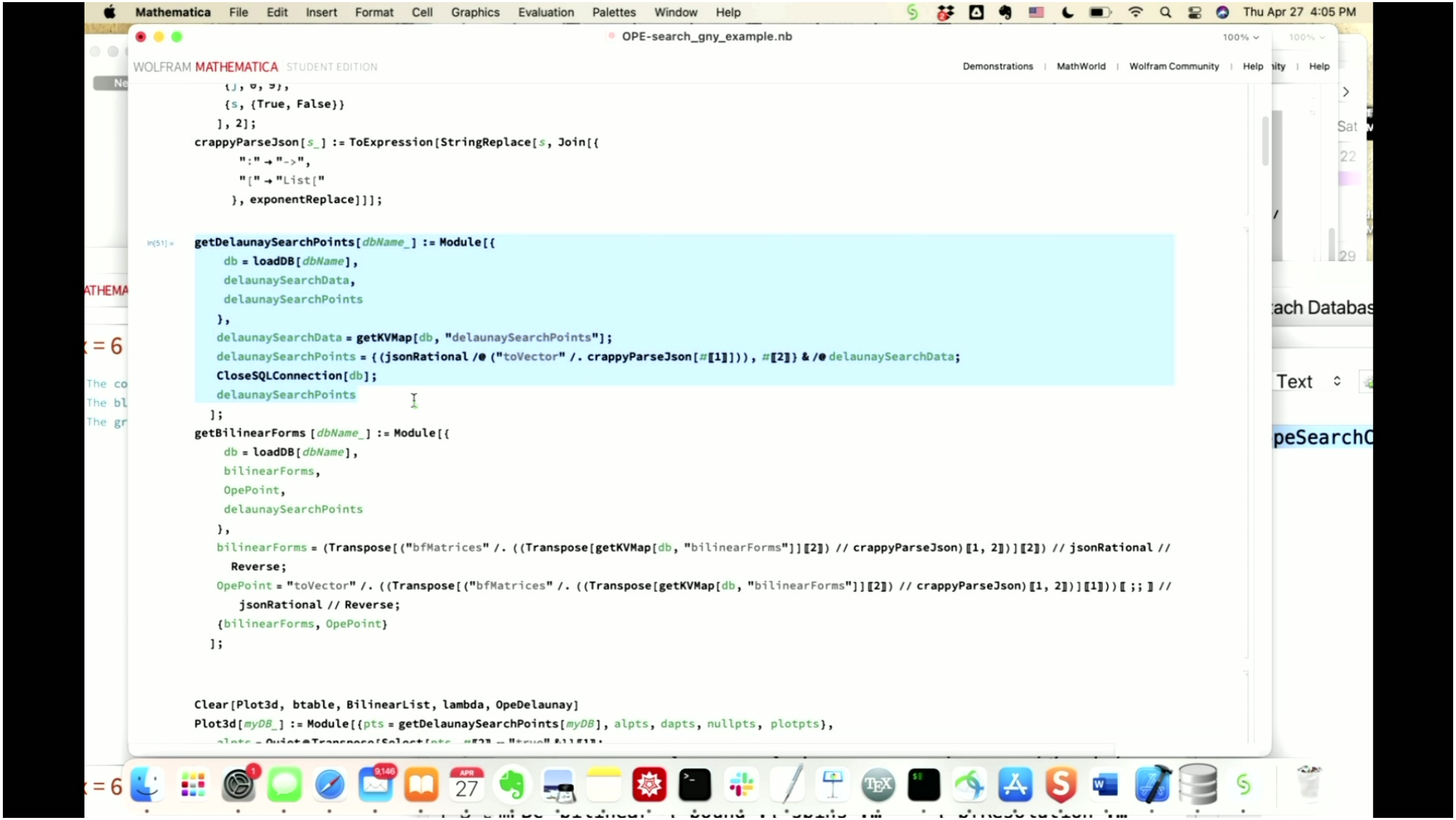
In[40]:=
```
Needs["DatabaseLink`"];
getKVMap[db_, kvMap_] := SQLExecute[db, "select key, val, max(created_at) from hyperion_key_val where kv_map = `1` group by key order by created_at", {kvMap}];
listKVMaps[db_] := Join @@ SQLExecute[db, "select kv_map from hyperion_key_val group by kv_map"];
loadDB[dbName_] := OpenSQLConnection[JDBC["SQLite", FileNameJoin[{NotebookDirectory[], dbName}]]];
Clear[jsonNumber];
jsonNumber[l_List] /; l[[1, 1]] == "$numberLong" := ToExpression[l[[1, 2]]];
jsonNumber[x_] := ToExpression[x];
Clear[jsonRational];
jsonRational[r_] := jsonNumber["numerator" /. r] / jsonNumber["denominator" /. r];
(* Deal with Mathematica's ridiculous notation for exponents *)
exponentReplace = Flatten[Table[
    StringJoin[ToString[i], "e", If[s, "-", ""], ToString[j]] → StringJoin[ToString[i], "*^", If[s, "-", ""], ToString[j]],
    {i, 0, 9},
    {j, 0, 9},
    {s, {True, False}}
   ], 2];
crappyParseJson[s_] := ToExpression[StringReplace[s, Join[{
      ":" → "->",
      "[" → "List["
     }, exponentReplace]]];
```

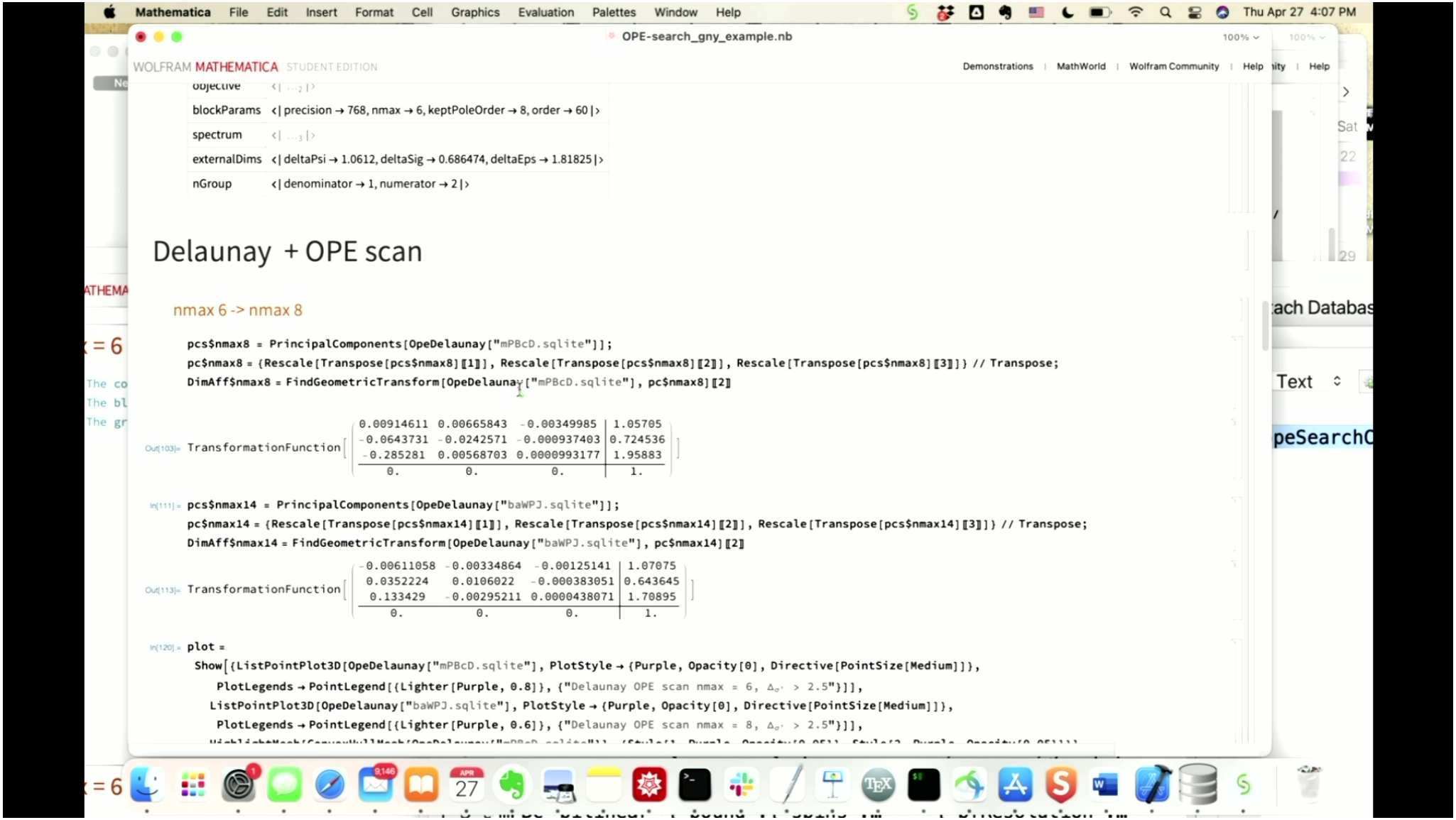
In[51]:=
```
getDelaunaySearchPoints[dbName_] := Module[{
    db = loadDB[dbName],
    delaunaySearchData,
    delaunaySearchPoints
   },
   delaunaySearchData = getKVMap[db, "delaunaySearchPoints"];
   delaunaySearchPoints = {(jsonRational /@ ("toVector" /. crappyParseJson[#[[1]]])), #[[2]]} & /@ delaunaySearchData;
```

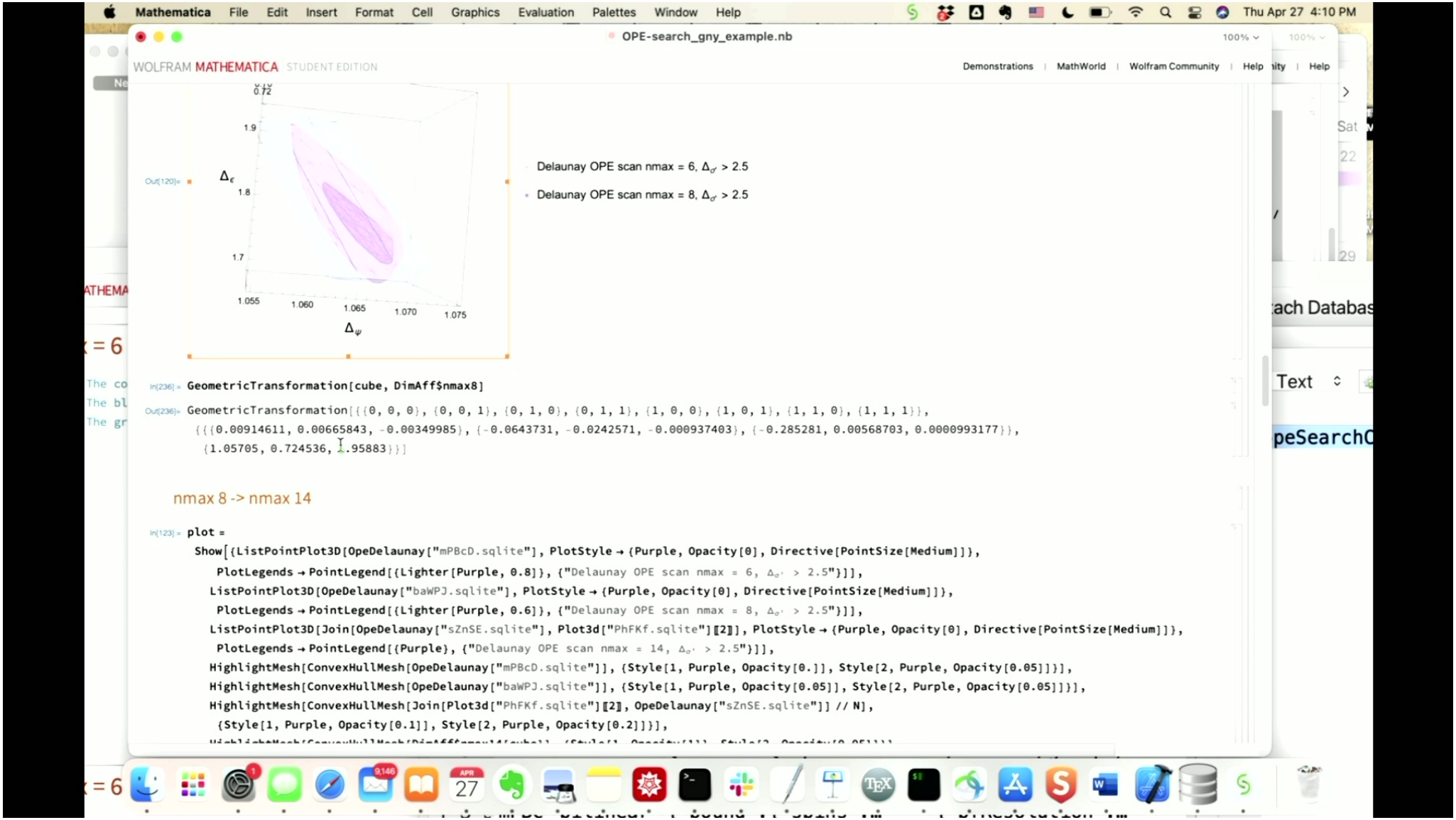

Mathematica File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
Thu Apr 27 4:01 PM
OPE-search_gny_example.nb
WOLFRAM MATHEMATICA | STUDENT EDITION
Demonstrations | MathWorld | Wolfram Community | Help
Out[38]= /Users/aikeliu/Downloads/sqlite-jdbc-3.36.0.3.jar
In[40]:=
Needs["DatabaseLink`"];
getKVMap[db_, kvMap_] :=
SQLExecute[db, "select key, val, max(created_at) from hyperion_key_val where kv_map = `1` group by key order by created_at", {kvMap}];
listKVMaps[db_] := Join @@ SQLExecute[db, "select kv_map from hyperion_key_val group by kv_map"];
loadDB[dbName_] := OpenSQLConnection[JDBC["SQLite", FileNameJoin[{NotebookDirectory[], dbName}]]];
Clear[jsonNumber];
jsonNumber[l_List] /; l[[1, 1]] == "$numberLong" := ToExpression[l[[1, 2]]];
jsonNumber[x_] := ToExpression[x];
Clear[jsonRational];
jsonRational[r_] := jsonNumber["numerator" /. r] / jsonNumber["denominator" /. r];
(* Deal with Mathematica's ridiculous notation for exponents *)
exponentReplace = Flatten[Table[
StringJoin[ToString[i], "e", If[s, "-", ""], ToString[j]] → StringJoin[ToString[i], "*^", If[s, "-", ""], ToString[j]],
{i, 0, 9},
{j, 0, 9},
{s, {True, False}}
], 2];
crappyParseJson[s_] := ToExpression[StringReplace[s, Join[{
":" → "->",
"[" → "List["
}, exponentReplace]]];
In[51]:=
getDelaunaySearchPoints[dbName_] := Module[{
db = loadDB[dbName],
delaunaySearchData,
delaunaySearchPoints
},
delaunaySearchData = getKVMap[db, "delaunaySearchPoints"];
delaunaySearchPoints = {(jsonRational /@ ("toVector" /. crappyParseJson[#[[1]]])), #[[2]]} & /@ delaunaySearchData;
CloseSQLConnection[db];
delaunaySearchPoints
1.80
1.75
1.70
1.65
1.60
1.55


Mathematica File Edit Ins... Form... Cell Graphics Evaluation Palett... Window Help
Thu Apr 27 4:02 PM
OPE-search_gny_example.nb
WOLFRAM MATHEMATICA STUDENT EDITION
Demonstrations | MathWorld | Wolfram Community | Help
Setup
In[36]:=
Needs["JLink`"]; Needs["DatabaseLink`" ;
ReinstallJava[] ;
jdbcdriver = FindFile["/Users/aikeliu/Downloads/sqlite-jdbc-3.36.0.3.jar"]
AddToClassPath[jdbcdriver, Prepend → True ;
Out[38]= /Users/aikeliu/Downloads/sqlite-jdbc-3.36.0.3.jar
In[40]:=
Needs["DatabaseLink`"];
getKVMap[db_, kvMap_] :=
SQLExecute[db, "select key, val, max(created_at) from hyperion_key_val where kv_map = ? group by key order by created_at", {kvMap}] ;
listKVMaps[db_] := JoinTo SQLExecute[db, "select kv_map from hyperion_key_val group by kv_map" ;
loadDB[dbName_] := OpenSQLConnection[JDBC["SQLite", FileNameJoin[{NotebookDirectory[], dbName}]]];
Clear[jsonNumber];
jsonNumber[l_List] /; l[[1, 1]] == "$numberLong" := ToExpression[l[[1, 2]]];
jsonNumber[x_] := ToExpression[x];
Clear[jsonRational]
jsonRational[r_] := jsonNumber["numerator" /. r] / jsonNumber["denominator" /. r] ;
(* Deal with Mathematica's ridiculous notation for exponents *)
exponentReplace = Flatten[Table[
StringJoin[ToString[i], "e", If[s, "-", ""], ToString[j]] → StringJoin[ToString[i], "*^", If[s, "-", ""], ToString[j]] ,
{i, 0, 9},
{j, 0, 9},
{s, {True, False}}
], 2 ;
crappyParseJson[s_] := ToExpression[StringReplace[s, Join[{
":" → "->",
"[" → "List["
}, exponentReplace]]];
1
1.75
1.70
1.65
1.60
1.55
APR
27


DB Browser for SQLite File Edit View Tools Help
Thu Apr 27 4:03 PM
DB Browser for SQLite - /Users/aikeliu/GNY/mPBcD.sqlite
New Database
Open Database
Open Project
Save Project
Attach Database
Close Database
Database Structure
Browse Data
Edit Pragmas
Execute SQL
Remote
Table: hyperion_key_
Edit Database Cell
Mode: Text
1 delaunaySearchPoints
Type of data currently in cell: Text / Numeric
20 character(s)
kv_map
key
1 mPBc programN "GNYSigPsiEps_delaunay
2 mPBc delaunay {"toVector":…
3 mPBc delaunay {"toVector":…
4 mPBc delaunay {"toVector":…
5 mPBc delaunay {"toVector":…
6 mPBc delaunay {"toVector":…
7 mPBc delaunay {"toVector":…
8 mPBc delaunay {"toVector":…
9 mPBc delaunay {"toVector":…
10 mPBc delaunay {"toVector":…
11 mPBc delaunay {"toVector":…
12 mPBc delaunay {"toVector":…
13 mPBc delaunay {"toVector":… true
14 mPBc delaunay {"toVector":… true
1 - 13 of 21346
Go to: 1
SQL Log
Plot
DB Schema
Remote


Mathematica File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
Thu Apr 27 4:05 PM
OPE-search_gny_example.nb
WOLFRAM MATHEMATICA STUDENT EDITION
Demonstrations | MathWorld | Wolfram Community | Help
{s, {True, False}}
], 2];
crappyParseJson[s_] := ToExpression[StringReplace[s, Join[{
":" → "->",
"[" → "List["
}, exponentReplace]]];
In[51]:=
getDelaunaySearchPoints[dbName_] := Module[{
db = loadDB[dbName],
delaunaySearchData,
delaunaySearchPoints
},
delaunaySearchData = getKVMap[db, "delaunaySearchPoints"];
delaunaySearchPoints = {(jsonRational /@ ("toVector" /. crappyParseJson[#[[1]]])), #[[2]]} & /@ delaunaySearchData;
CloseSQLConnection[db];
delaunaySearchPoints
];
getBilinearForms [dbName_] := Module[{
db = loadDB[dbName],
bilinearForms,
OpePoint,
delaunaySearchPoints
},
bilinearForms = (Transpose[("bfMatrices" /. ((Transpose[getKVMap[db, "bilinearForms"]][[2]]) // crappyParseJson)[[1, 2]])][[2]]) // jsonRational //
Reverse;
OpePoint = "toVector" /. ((Transpose[("bfMatrices" /. ((Transpose[getKVMap[db, "bilinearForms"]][[2]]) // crappyParseJson)[[1, 2]])][[1]]))[[;;]] //
jsonRational // Reverse;
{bilinearForms, OpePoint}
];
Clear[Plot3d, btable, BilinearList, lambda, OpeDelaunay]
Plot3d[myDB_] := Module[{pts = getDelaunaySearchPoints[myDB], alpts, dapts, nullpts, plotpts},


Mathematica File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
Thu Apr 27 4:07 PM
OPE-search_gny_example.nb
WOLFRAM MATHEMATICA STUDENT EDITION
Demonstrations | MathWorld | Wolfram Community | Help
blockParams <| precision → 768, nmax → 6, keptPoleOrder → 8, order → 60 |>
spectrum <| ... |>
externalDims <| deltaPsi → 1.0612, deltaSig → 0.686474, deltaEps → 1.81825 |>
nGroup <| denominator → 1, numerator → 2 |>
Delaunay + OPE scan
nmax 6 -> nmax 8
pcs$nmax8 = PrincipalComponents[OpeDelaunay["mPBcD.sqlite"]];
pc$nmax8 = {Rescale[Transpose[pcs$nmax8][[1]]], Rescale[Transpose[pcs$nmax8][[2]]], Rescale[Transpose[pcs$nmax8][[3]]]} // Transpose;
DimAff$nmax8 = FindGeometricTransform[OpeDelaunay["mPBcD.sqlite"], pc$nmax8][[2]]
Out[103]= TransformationFunction
0.00914611 0.00665843 -0.00349985 | 1.05705
-0.0643731 -0.0242571 -0.000937403 | 0.724536
-0.285281 0.00568703 0.0000993177 | 1.95883
0. 0. 0. | 1.
In[111]= pcs$nmax14 = PrincipalComponents[OpeDelaunay["baWPJ.sqlite"]];
pc$nmax14 = {Rescale[Transpose[pcs$nmax14][[1]]], Rescale[Transpose[pcs$nmax14][[2]]], Rescale[Transpose[pcs$nmax14][[3]]]} // Transpose;
DimAff$nmax14 = FindGeometricTransform[OpeDelaunay["baWPJ.sqlite"], pc$nmax14][[2]]
Out[113]= TransformationFunction
-0.00611058 -0.00334864 -0.00125141 | 1.07075
0.0352224 0.0106022 -0.000383051 | 0.643645
0.133429 -0.00295211 0.0000438071 | 1.70895
0. 0. 0. | 1.
In[120]= plot =
Show[{ListPointPlot3D[OpeDelaunay["mPBcD.sqlite"], PlotStyle → {Purple, Opacity[0], Directive[PointSize[Medium]]},
PlotLegends → PointLegend[{Lighter[Purple, 0.8]}, {"Delaunay OPE scan nmax = 6, Δσ' > 2.5"}]],
ListPointPlot3D[OpeDelaunay["baWPJ.sqlite"], PlotStyle → {Purple, Opacity[0], Directive[PointSize[Medium]]},
PlotLegends → PointLegend[{Lighter[Purple, 0.6]}, {"Delaunay OPE scan nmax = 8, Δσ' > 2.5"}]],


Mathematica File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
Thu Apr 27 4:10 PM
OPE-search_gny_example.nb
WOLFRAM MATHEMATICA STUDENT EDITION
Demonstrations | MathWorld | Wolfram Community | Help
Out[120]=
Δ_ϵ
Δ_ψ
1.9
1.8
1.7
1.055
1.060
1.065
1.070
1.075
Delaunay OPE scan nmax = 6, $\Delta_{\sigma'} > 2.5$
Delaunay OPE scan nmax = 8, $\Delta_{\sigma'} > 2.5$
In[236]= GeometricTransformation[cube, DimAff$nmax8]
Out[236]= GeometricTransformation[{{0, 0, 0}, {0, 0, 1}, {0, 1, 0}, {0, 1, 1}, {1, 0, 0}, {1, 0, 1}, {1, 1, 0}, {1, 1, 1}},
{{{0.00914611, 0.00665843, -0.00349985}, {-0.0643731, -0.0242571, -0.000937403}, {-0.285281, 0.00568703, 0.0000993177}},
{1.05705, 0.724536, 1.95883}}]
nmax 8 -> nmax 14
In[123]= plot =
Show[{ListPointPlot3D[OpeDelaunay["mPBcD.sqlite"], PlotStyle → {Purple, Opacity[0], Directive[PointSize[Medium]]},
PlotLegends → PointLegend[{Lighter[Purple, 0.8]}, {"Delaunay OPE scan nmax = 6, $\Delta_{\sigma'} > 2.5$"}]],
ListPointPlot3D[OpeDelaunay["baWPJ.sqlite"], PlotStyle → {Purple, Opacity[0], Directive[PointSize[Medium]]},
PlotLegends → PointLegend[{Lighter[Purple, 0.6]}, {"Delaunay OPE scan nmax = 8, $\Delta_{\sigma'} > 2.5$"}]],
ListPointPlot3D[Join[OpeDelaunay["sZnSE.sqlite"], Plot3d["PhFKf.sqlite"][[2]]], PlotStyle → {Purple, Opacity[0], Directive[PointSize[Medium]]},
PlotLegends → PointLegend[{Purple}, {"Delaunay OPE scan nmax = 14, $\Delta_{\sigma'} > 2.5$"}]],
HighlightMesh[ConvexHullMesh[OpeDelaunay["mPBcD.sqlite"]], {Style[1, Purple, Opacity[0.]], Style[2, Purple, Opacity[0.05]]}],
HighlightMesh[ConvexHullMesh[OpeDelaunay["baWPJ.sqlite"]], {Style[1, Purple, Opacity[0.05]], Style[2, Purple, Opacity[0.05]]}],
HighlightMesh[ConvexHullMesh[Join[Plot3d["PhFKf.sqlite"][[2]], OpeDelaunay["sZnSE.sqlite"]] // N],
{Style[1, Purple, Opacity[0.1]], Style[2, Purple, Opacity[0.2]]}],

Mathematica File Edit Insert Format Cell Graphics Evaluation Palettes Window Help

Thu Apr 27 4:11 PM

OPE-search_gny_example.nb

WOLFRAM MATHEMATICA STUDENT EDITION

Demonstrations | MathWorld | Wolfram Community | Help | Help

OPE Space (nmax 6 -> nmax 8)

```
pcs$OPE$nmax8 = PrincipalComponents[lambda3d["mPBcD.sqlite"]];
pc$OPE$nmax8 = {Rescale[Transpose[pcs$OPE$nmax8][[1]]], Rescale[Transpose[pcs$OPE$nmax8][[2]]], Rescale[Transpose[pcs$OPE$nmax8][[3]]]} // Transpose;
OpeAff$nmax8 = FindGeometricTransform[lambda3d["mPBcD.sqlite"], pc$OPE$nmax8][[2]]
Inv$OpeAff$nmax8 = FindGeometricTransform[pc$OPE$nmax8, lambda3d["mPBcD.sqlite"]][[2]]
Show[ListPointPlot3D[pc$OPE$nmax8, PlotRange → {{-1.5, 2.5}, {-1.5, 2.5}, {-1.5, 2.5}}, PlotStyle → {Green, Opacity[1]}],
 RegionPlot3D[(x - 1/2)^2 + (y - 1/2)^2 + (z - 1/2)^2 - 4, {x, -1.5, 2.5}, {y, -1.5, 2.5}, {z, -1.5, 2.5}, AxesLabel → Automatic,
  PlotStyle → {2, LightBlue, Opacity[0.6]}, MeshStyle → {{Opacity[0.1], Thin}, {Opacity[0.1], Thin}}, PlotStyle → {1, Opacity[0.]},
  PlotLegends → PointLegend[{Automatic}, {"Binding ellipse, r = 2"}]], BoxRatios → {1, 1, 1}]
```

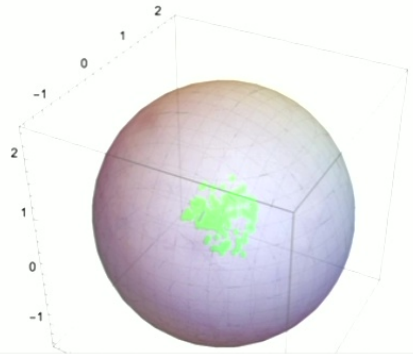
Out[141]=

$$\text{TransformationFunction}\left[\left(\begin{array}{ccc|c} 0.07899 & 0.000439527 & -0.0106161 & 0.275035 \\ 0.0490523 & 0.0201954 & 0.00140371 & 0.103014 \\ 0.182552 & -0.00561676 & 0.0042164 & 0.525326 \\ \hline 0. & 0. & 0. & 1. \end{array}\right)\right]$$

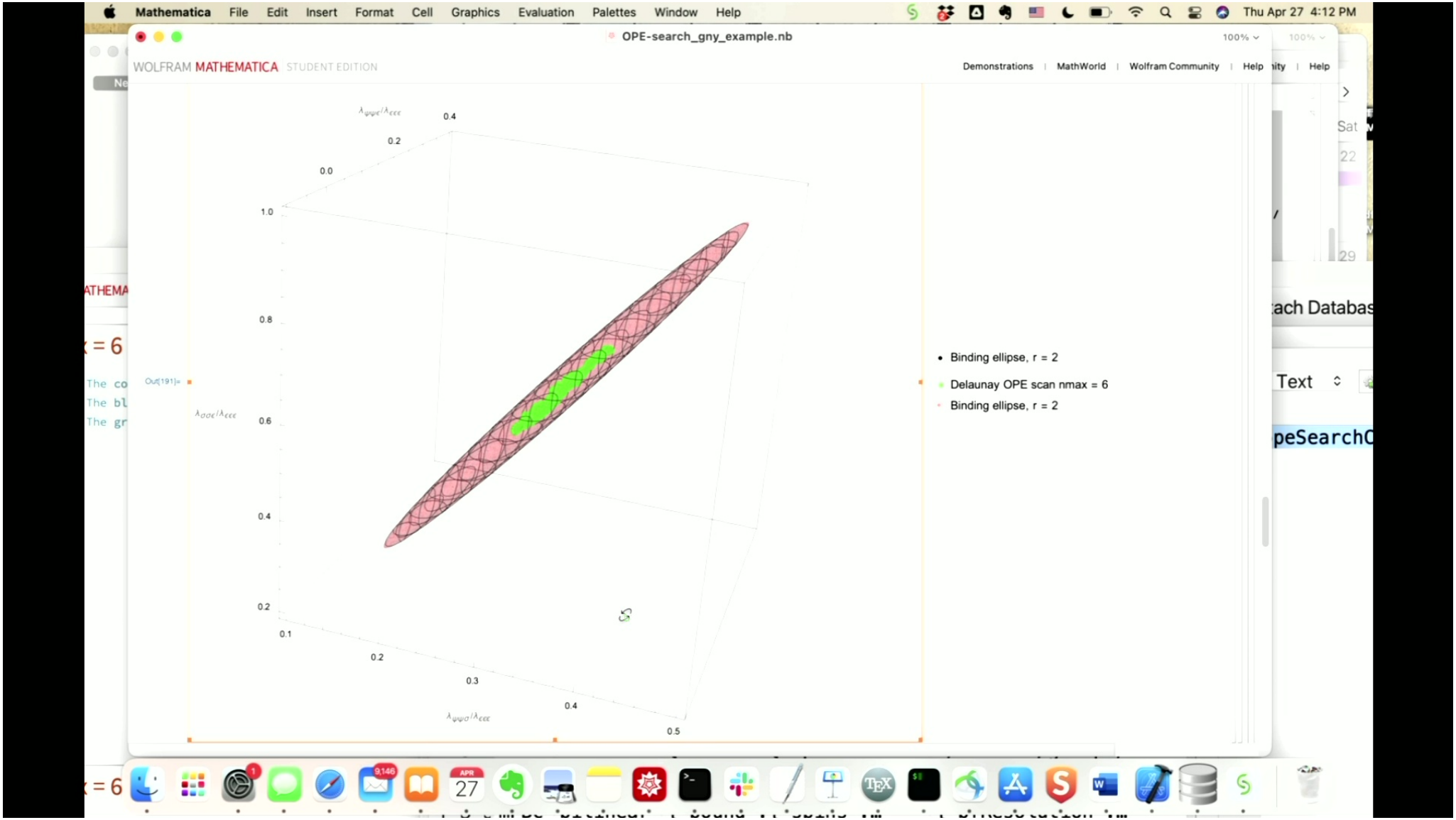
Out[142]=

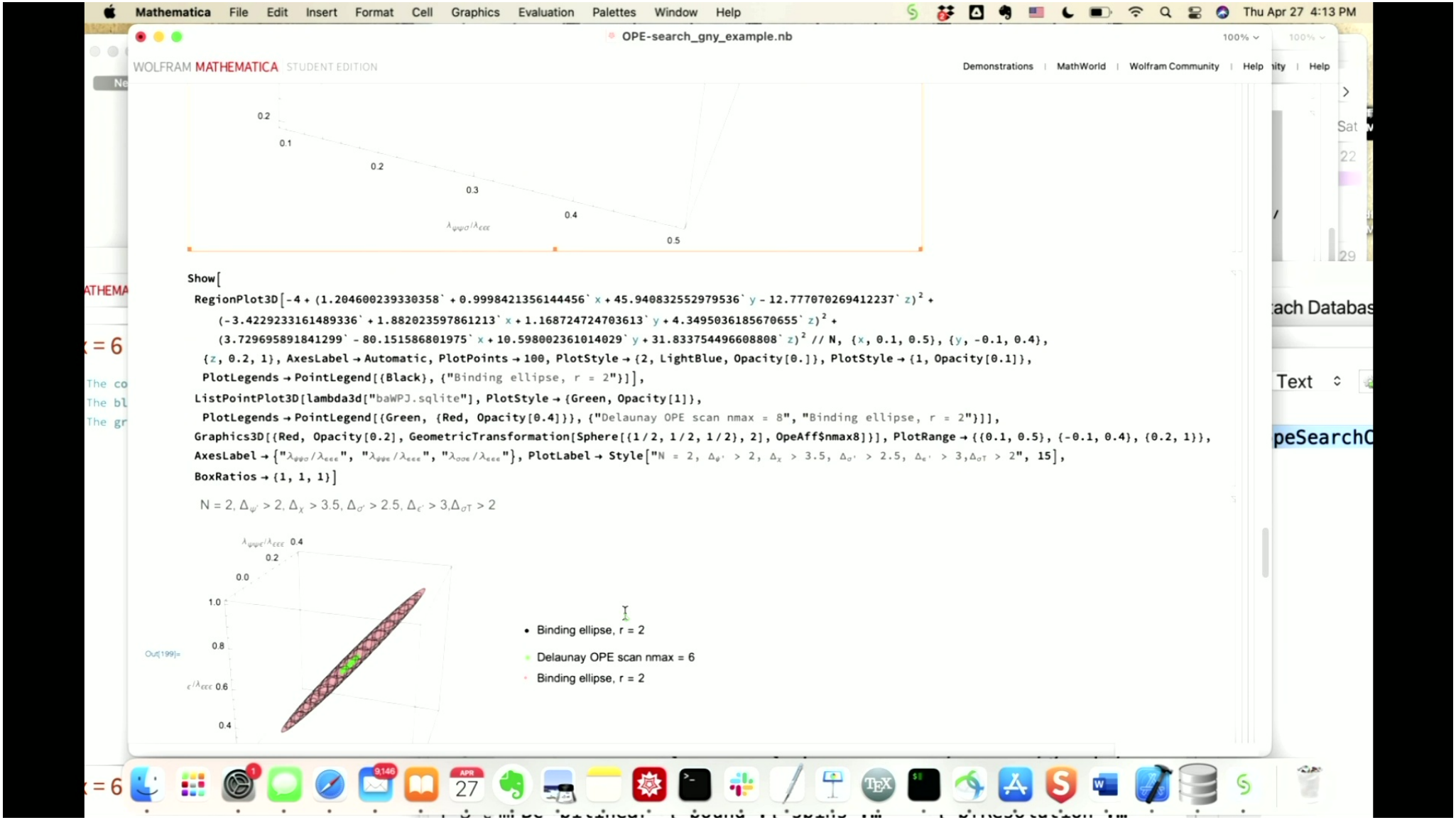
$$\text{TransformationFunction}\left[\left(\begin{array}{ccc|c} 1.88202 & 1.16872 & 4.3495 & -2.92292 \\ 0.999842 & 45.9408 & -12.7771 & 1.7046 \\ -80.1516 & 10.598 & 31.8338 & 4.2297 \\ \hline 0. & 0. & 0. & 1. \end{array}\right)\right]$$

Out[143]=



- Binding ellipse, r = 2


Mathematica File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
Thu Apr 27 4:12 PM
OPE-search_gny_example.nb
WOLFRAM MATHEMATICA STUDENT EDITION
Demonstrations | MathWorld | Wolfram Community | Help
Out[191]=
Binding ellipse, r = 2
Delaunay OPE scan nmax = 6
Binding ellipse, r = 2


Mathematica File Edit Insert Format Cell Graphics Evaluation Palettes Window Help
Thu Apr 27 4:13 PM
OPE-search_gny_example.nb
WOLFRAM MATHEMATICA STUDENT EDITION
Demonstrations | MathWorld | Wolfram Community | Help
0.2
0.1
0.2
0.3
0.4
0.5
Show[
RegionPlot3D[-4 + (1.204600239330358` + 0.9998421356144456` x + 45.940832552979536` y - 12.777070269412237` z)^2 +
(-3.4229233161489336` + 1.882023597861213` x + 1.168724724703613` y + 4.3495036185670655` z)^2 +
(3.729695891841299` - 80.151586801975` x + 10.598002361014029` y + 31.833754496608808` z)^2 // N, {x, 0.1, 0.5}, {y, -0.1, 0.4},
{z, 0.2, 1}, AxesLabel → Automatic, PlotPoints → 100, PlotStyle → {2, LightBlue, Opacity[0.]}, PlotStyle → {1, Opacity[0.1]},
PlotLegends → PointLegend[{Black}, {"Binding ellipse, r = 2"}]],
ListPointPlot3D[lambda3d["baWPJ.sqlite"], PlotStyle → {Green, Opacity[1]},
PlotLegends → PointLegend[{Green, {Red, Opacity[0.4]}}, {"Delaunay OPE scan nmax = 8", "Binding ellipse, r = 2"}]],
Graphics3D[{Red, Opacity[0.2], GeometricTransformation[Sphere[{1/2, 1/2, 1/2}, 2], OpeAff$nmax8]}], PlotRange → {{0.1, 0.5}, {-0.1, 0.4}, {0.2, 1}},
AxesLabel → {"λψψσ/λεεε", "λψψε/λεεε", "λσσε/λεεε"}, PlotLabel → Style["N = 2, Δψ' > 2, Δχ > 3.5, Δσ' > 2.5, Δε' > 3,ΔσT > 2", 15],
BoxRatios → {1, 1, 1}]
N = 2, Δψ' > 2, Δχ > 3.5, Δσ' > 2.5, Δε' > 3,ΔσT > 2
Out[199]=
Binding ellipse, r = 2
Delaunay OPE scan nmax = 6
Binding ellipse, r = 2