```
Thread 1 "netgen" hit Breakpoint 1, ngcomp::GridFunctionCoefficientFunction::Dimension
(this=0x15454b0)
    at /home/dow/ngsuite/ngsolve-src/comp/gridfunction.cpp:1081
1081        throw Exception(string ("don't know my dimension, space is ") +
(gdb) where

#0  ngcomp::GridFunctionCoefficientFunction::Dimension (this=0x15454b0)
    at /home/dow/ngsuite/ngsolve-src/comp/gridfunction.cpp:1081
```

```
  int GridFunctionCoefficientFunction::Dimension() const
  {
    for (auto vb : { VOL, BND, BBND })
      if (diffop[vb])
        return diffop[vb]->Dim();
    /*
    if (diffop) return diffop->Dim();
    if (trace_diffop) return trace_diffop->Dim();
    if (bfi) return bfi->DimFlux();
    if (gf->GetFESpace()->GetEvaluator())
      return gf->GetFESpace()->GetEvaluator()->Dim();
    */
    throw Exception(string ("don't know my dimension, space is ") +
                    typeid(*gf->GetFESpace()).name());
  }
```

```
#1  0x00007fffe8bc2c0b in ngcomp::Visualize (
    gf=std::shared_ptr (count 2, weak 1) 0x1545160, given_name="euq")
    at /home/dow/ngsuite/ngsolve-src/comp/gridfunction.cpp:250
```

```
  // void GridFunction :: Visualize(const string & given_name)
  void Visualize(shared_ptr<GridFunction> gf, const string & given_name)
  {
    auto fespace = gf->GetFESpace();
    auto ma = fespace->GetMeshAccess();

    shared_ptr<DifferentialOperator> eval_2d, eval_3d;
    if (ma->GetDimension() == 2)
      {
        eval_2d = fespace->GetEvaluator(VOL);
      }
    else
      {
        eval_3d = fespace->GetEvaluator(VOL);
        eval_2d = fespace->GetEvaluator(BND);
      }


    netgen::SolutionData * vis = new VisualizeCoefficientFunction (ma, gf);
    Ng_SolutionData soldata;
    Ng_InitSolutionData (&soldata);

    soldata.name = given_name;
    soldata.data = 0;
    soldata.components = gf -> Dimension();
```

```
#2  0x00007fffe8bffcaa in ngcomp::T_GridFunction<std::complex<double> >::T_GridFunction
(this=this@entry=0x1545160,
    afespace=<error reading variable: access outside bounds of object referenced via synthetic pointer>,
aname="euq", flags=..., __in_chrg=<optimized out>,
    __vtt_parm=<optimized out>)
    at /home/dow/ngsuite/ngsolve-src/comp/gridfunction.cpp:858
```

```
  template <class SCAL>
  S_ComponentGridFunction<SCAL> ::
  S_ComponentGridFunction (const S_GridFunction<SCAL> & agf_parent, int acomp)
    : S_GridFunction<SCAL> (dynamic_cast<const CompoundFESpace&>
(*agf_parent.GetFESpace())[acomp],
                                        agf_parent.GetName()+"."+ToString (acomp+1), Flags()),
      gf_parent(agf_parent), comp(acomp)
  {
    this->SetVisual(agf_parent.GetVisual());
    const CompoundFESpace * cfe = dynamic_cast<const CompoundFESpace *>(this-
>GetFESpace().get());
    if (cfe)
      {
        int nsp = cfe->GetNSpaces();
        this->compgfs.SetSize(nsp);
        for (int i = 0; i < nsp; i++)
          this->compgfs[i] = make_shared<S_ComponentGridFunction<SCAL>> (*this, i);
      }

    // this->Visualize (this->name);
    if (this->visual)
      Visualize (shared_ptr<GridFunction> (this, NOOP_Deleter), this->name);
  }

  template <class SCAL>
  S_ComponentGridFunction<SCAL> ::
  ~S_ComponentGridFunction ()
  {
    this -> vec = NULL;  // base-class desctructor must not delete the vector
  }


  template <class SCAL>
  void S_ComponentGridFunction<SCAL> :: Update()
  {
    const CompoundFESpace & cfes = dynamic_cast<const CompoundFESpace&> (*gf_parent.GetFESpace().get());

    this -> vec.SetSize (gf_parent.GetMultiDim());
    GridFunction::multidim = gf_parent.GetMultiDim();
    for (int i = 0; i < gf_parent.GetMultiDim(); i++)
      (this->vec)[i] = gf_parent.GetVector(i).Range (cfes.GetRange(comp));

    this -> level_updated = this -> ma->GetNLevels();

    for (int i = 0; i < this->compgfs.Size(); i++)
      this->compgfs[i]->Update();
  }

  template <class TV>
  T_GridFunction<TV> ::
  T_GridFunction (const FESpace & afespace, const string & aname, const Flags & flags)
    : T_GridFunction(shared_ptr<FESpace> (const_cast<FESpace*>(&afespace),NOOP_Deleter), aname, flags)
  { ; }

  template <class TV>
  T_GridFunction<TV> ::
  T_GridFunction (shared_ptr<FESpace> afespace, const string & aname, const Flags & flags)
    : S_GridFunction<TSCAL> (afespace, aname, flags)
  {
    vec.SetSize (this->multidim);
    vec = 0;

    const CompoundFESpace * cfe = dynamic_cast<const CompoundFESpace *>(this->GetFESpace().get());
    if (cfe)
      {
            int nsp = cfe->GetNSpaces();
            compgfs.SetSize(nsp);
            for (int i = 0; i < nsp; i++)
              compgfs[i] = make_shared<S_ComponentGridFunction<TSCAL>> (*this, i);
      }

    // this->Visualize (this->name);
    if (this->visual)
      Visualize(shared_ptr<GridFunction> (this, NOOP_Deleter), this->name);
```