MECHANIZED MATHEMATICS AND ITS APPLICATIONS, VOL. 6, NO. 1, NOVEMBER 2007, 1:14

On the Formalization of Linear Operators

Noboru Endou *Yasunari Shidama*

Shinshu University, Faculty of Engineering
Nagano-ken Nagano-shi Wakasato 4-17-1 380-8553 Japan
endon@gifu-nct.ac.jp, shidama@cs.shinshu-u.ac.jp

Abstract – In this paper, we report the progress of our work on the development of a library on linear operators within the Mizar project. The definitions of concepts including normed linear spaces of bounded linear operators and completeness of such spaces are contained in this library.

1. Introduction

Until the 1970s, research on linear operators held a central role in the work of functional analysis and today its importance is unchanged. Linear operators appear not only in pure mathematics, but they also serve important roles in a wide range of areas of applied mathematics, physics, and engineering. For example, in engineering control theory, linear control theory by Kalman was developed by taking the theory of matrices, the most important example of bounded linear operators, and expanding it to control theory with developing systems of equations by Lions which uses unbounded partial differentiation operators. In this work, the authors are actively involved in the contruction of Mizar library articles concerning analysis. They have contributed the LOPBAN series articles [1]–[4] and the versions for complex numbers in the CLOPBAN series [5]–[8] to construct a library for handling linear structures and normed structures of spaces generated by families of bounded (continuous) linear operators. The present objective will be to continue formalizing material in this area until the Yoshida-Hille C_0 semi-groups are covered. This paper outlines the formalization and reports the work completed thus far.

2. Outline of Formalization

2.1 Spaces Created by Bounded Linear Operators

It is known that mappings f, g from a set X to a space Y having a real linear space structure can be defined. We can also define for a real number r, addition $f+g$ and zero dimensional scalar multiplication rf operations. We begin with the formalization of these concepts.

```
definition
  let X be set;
  let Y be non empty set;
  let F be Function of [:REAL, Y:], Y;
  let a be real number, f be Function of X, Y;
  redefine func F[;](a,f) -> Element of Funcs(X, Y);
end;
```

Manuscript received May 30, 2007; revised July 21, 2007.

```
theorem :: LOPBAN_1:1
for X be non empty set for Y be non empty LoopStr
ex ADD be BinOp of Funcs(X,the carrier of Y) st
  for f,g being Element of Funcs(X,the carrier of Y)
    holds ADD.(f,g)=(the add of Y).:(f,g);

theorem :: LOPBAN_1:2
for X be non empty set, Y be RealLinearSpace
ex MULT be Function of
  [:REAL, Funcs(X,the carrier of Y):], Funcs(X,the carrier of Y) st
    for r be Real, f be Element of Funcs(X,the carrier of Y) holds
     for s be Element of X holds (MULT.[r,f]).s = r*(f.s);

definition
  let X be non empty set;
  let Y be non empty LoopStr;
  func FuncAdd(X,Y) -> BinOp of Funcs(X,the carrier of Y) means
:: LOPBAN_1:def 1
    for f,g being Element of Funcs(X,the carrier of Y) holds
    it.(f,g) = (the add of Y).:(f,g);
end;

definition
  let X be non empty set;
  let Y be RealLinearSpace;
  func FuncExtMult(X,Y) -> Function of [:REAL,Funcs(X,the carrier of Y):],
     Funcs(X,the carrier of Y) means
:: LOPBAN_1:def 2
     for a being Real,
    f being Element of Funcs(X,the carrier of Y),
      x being Element of X holds
      (it.[a,f]).x = a*(f.x);
end;

definition
  let X be set;
  let Y be non empty ZeroStr;
  func FuncZero(X,Y) -> Element of Funcs (X,the carrier of Y) equals
:: LOPBAN_1:def 3
   X --> 0.Y;
end;
```

Based on the definitions of these operations, we introduced a functor for expressing real linear spaces created by the entire mapping X to Y as shown below.

```
definition
  let X be non empty set;
  let Y be RealLinearSpace;
  func RealVectSpace(X,Y) -> RealLinearSpace equals
:: LOPBAN_1:def 4
   RLSStruct(#Funcs(X,the carrier of Y),
              (FuncZero(X,Y)),FuncAdd(X,Y),FuncExtMult(X,Y)#);
end;


theorem :: LOPBAN_1:14
  for X be non empty set
  for Y be RealLinearSpace
  for f,g,h be VECTOR of RealVectSpace(X,Y) holds
      h = f+g iff for x be Element of X holds h.x = f.x + g.x;

theorem :: LOPBAN_1:15
  for X be non empty set
  for Y be RealLinearSpace
  for f,h be VECTOR of RealVectSpace(X,Y)
  for a be Real holds
      h = a*f iff for x be Element of X holds h.x = a * f.x;

theorem :: LOPBAN_1:16
  for X be non empty set
  for Y be RealLinearSpace holds  0.RealVectSpace(X,Y) = X --> 0.Y;
```

For a mapping f from a space X to Y with real linear space structures, when we have linearity of addition, the *additive* attribute, and homogeniety of scalar multiplication, the *homogeneous* attribute, we call this a linear mapping. In the following, we show the formalization of these mapping attributes and introduce them as type variables, or modes, for linear operators.

```
definition
  let X be non empty RLSStruct;
  let Y be non empty LoopStr;
  let IT be Function of X, Y;
  attr IT is additive means
:: LOPBAN_1:def 5
  for x,y being VECTOR of X holds IT.(x+y) = IT.x+IT.y;
```

```
end;

definition
  let X, Y be non empty RLSStruct;
  let IT be Function of X,Y;
  attr IT is homogeneous means
:: LOPBAN_1:def 6
   for x being VECTOR of X, r being Real holds IT.(r*x) = r*IT.x;
end;

registration
  let X be non empty RLSStruct;
  let Y be RealLinearSpace;
  cluster additive homogeneous Function of X,Y;
end;

definition
  let X, Y be RealLinearSpace;
  mode LinearOperator of X,Y is additive homogeneous Function of X,Y;
end;
```

To treat the set of all linear operators, we introduce a functor called LinearOperators formalized as follows.

```
definition
  let X, Y be RealLinearSpace;
  func LinearOperators(X,Y) -> Subset of
     RealVectSpace(the carrier of X, Y) means
:: LOPBAN_1:def 7
   for x being set holds x in it
     iff
   x is LinearOperator of X,Y;
end;
```

Since $LinearOperators(X,Y)$ forms a linearly closed subset of the real linear space $RealVectSpace(X,Y)$ generated by the entire mapping of X to Y mentioned earlier, we can introduce a real linear space structure, a subspace of $RealVectSpace(X,Y)$, which uses the set of all linear operators from real linear space X to Y as a base set. The formalization of this procedure is shown below.

We begin by introducing a real linear space using a functor called $R\_VectorSpace\_of\_LinearOperators(X,Y)$.

```
theorem :: LOPBAN_1:17
  for X, Y be RealLinearSpace holds LinearOperators(X,Y) is lineary-closed;
```

```
theorem :: LOPBAN_1:18
  for X, Y be RealLinearSpace holds
RLSStruct (# LinearOperators(X,Y),
        Zero_(LinearOperators(X,Y),RealVectSpace(the carrier of X,Y)),
        Add_(LinearOperators(X,Y), RealVectSpace(the carrier of X,Y)),
        Mult_(LinearOperators(X,Y), RealVectSpace(the carrier of X,Y)) #)
is Subspace of RealVectSpace(the carrier of X,Y);

registration
  let X, Y be RealLinearSpace;
  cluster RLSStruct (# LinearOperators(X,Y),
        Zero_(LinearOperators(X,Y),RealVectSpace(the carrier of X,Y)),
        Add_(LinearOperators(X,Y), RealVectSpace(the carrier of X,Y)),
        Mult_(LinearOperators(X,Y), RealVectSpace(the carrier of X,Y)) #)
-> Abelian add-associative right_zeroed right_complementable
        RealLinearSpace-like;
end;

definition
  let X, Y be RealLinearSpace;
  func R_VectorSpace_of_LinearOperators(X,Y) -> RealLinearSpace equals
:: LOPBAN_1:def 8
  RLSStruct (# LinearOperators(X,Y),
             Zero_(LinearOperators(X,Y),RealVectSpace(the carrier of X,Y)),
           Add_(LinearOperators(X,Y), RealVectSpace(the carrier of X,Y)),
           Mult_(LinearOperators(X,Y), RealVectSpace(the carrier of X,Y))
#);
end;
```

Next, we define the boundedness of linear operators as an attribute in the same way as linearity. We introduce the set of all bounded linear operators as a functor.

```
definition
  let X, Y be RealNormSpace;
  let IT be LinearOperator of X,Y;
  attr IT is bounded means
:: LOPBAN_1:def 9
  ex K being Real st 0 <= K &
   for x being VECTOR of X holds
       ||. IT.x .|| <= K * ||. x .||;
end;

definition
  let X, Y be RealNormSpace;
```

```
  func BoundedLinearOperators(X,Y) ->
    Subset of R_VectorSpace_of_LinearOperators(X,Y) means
:: LOPBAN_1:def 10
   for x being set holds x in it
     iff
   x is bounded LinearOperator of X,Y;
end;
```

Since this forms a closed subset of $R\_VectorSpace\_of\_LinearOperators(X,Y)$ introduced above, we can repeat the procedure mentioned earlier and introduce a real linear space structure which uses the set of all bounded linear operators from real linear space X to Y as a base set in a similar way. The steps for this formalization are shown below.

We introduce a real linear space using a functor called $R\_VectorSpace\_of\_BoundedLinearOperators(X,Y)$.

```
theorem :: LOPBAN_1:26
  for X, Y be RealNormSpace holds
  BoundedLinearOperators(X,Y) is lineary-closed;

theorem :: LOPBAN_1:27
  for X, Y be RealNormSpace holds
RLSStruct (# BoundedLinearOperators(X,Y),
        Zero_(BoundedLinearOperators(X,Y),
        R_VectorSpace_of_LinearOperators(X,Y)),
         Add_(BoundedLinearOperators(X,Y),
         R_VectorSpace_of_LinearOperators(X,Y)),
        Mult_(BoundedLinearOperators(X,Y),
        R_VectorSpace_of_LinearOperators(X,Y)) #)
is Subspace of R_VectorSpace_of_LinearOperators(X,Y);

registration
  let X, Y be RealNormSpace;
  cluster RLSStruct (# BoundedLinearOperators(X,Y),
        Zero_(BoundedLinearOperators(X,Y),
        R_VectorSpace_of_LinearOperators(X,Y)),
         Add_(BoundedLinearOperators(X,Y),
         R_VectorSpace_of_LinearOperators(X,Y)),
        Mult_(BoundedLinearOperators(X,Y),
        R_VectorSpace_of_LinearOperators(X,Y)) #)
-> Abelian add-associative right_zeroed right_complementable
        RealLinearSpace-like;
end;
```

```
definition
  let X, Y be RealNormSpace;
  func R_VectorSpace_of_BoundedLinearOperators(X,Y) -> RealLinearSpace
equals
:: LOPBAN_1:def 11

  RLSStruct (# BoundedLinearOperators(X,Y),
        Zero_(BoundedLinearOperators(X,Y),
        R_VectorSpace_of_LinearOperators(X,Y)),
         Add_(BoundedLinearOperators(X,Y),
         R_VectorSpace_of_LinearOperators(X,Y)),
        Mult_(BoundedLinearOperators(X,Y),
        R_VectorSpace_of_LinearOperators(X,Y)) #);
end;
```

It is known that we can define a norm for bounded linear operators u and we formalize it below.

```
definition
   let X, Y be RealNormSpace;
   let u be LinearOperator of X,Y;
   func PreNorms(u) -> non empty Subset of REAL equals
:: LOPBAN_1:def 13
  {||.u.t.|| where t is VECTOR of X : ||.t.|| <= 1 };
end;

theorem :: LOPBAN_1:32
    for X, Y be RealNormSpace
    for g be bounded LinearOperator of X,Y
      holds PreNorms(g) is non empty bounded_above;

theorem :: LOPBAN_1:33
      for X, Y be RealNormSpace
    for g be LinearOperator of X,Y
      holds g is bounded iff PreNorms(g) is bounded_above;

theorem :: LOPBAN_1:34
   for X, Y be RealNormSpace
    ex NORM be Function of BoundedLinearOperators(X,Y),REAL st
          for f be set st f in BoundedLinearOperators(X,Y) holds
        NORM.f = sup PreNorms(modetrans(f,X,Y));

definition
   let X, Y be RealNormSpace;
   func BoundedLinearOperatorsNorm(X,Y)
```

```
     -> Function of BoundedLinearOperators(X,Y), REAL means
:: LOPBAN_1:def 14
  for x be set st x in BoundedLinearOperators(X,Y) holds
    it.x = sup PreNorms(modetrans(x,X,Y));
end;

theorem :: LOPBAN_1:35
   for X, Y be RealNormSpace
   for f be bounded LinearOperator of X,Y holds modetrans(f,X,Y)=f;

theorem :: LOPBAN_1:36
   for X, Y be RealNormSpace
   for f be bounded LinearOperator of X,Y holds
   BoundedLinearOperatorsNorm(X,Y).f = sup PreNorms(f);
```

From the formalization steps above, we can define a real normed linear space structure which uses the set of all bounded linear operators from real linear space X to Y as a base set. It is introduced as a functor

$$R\_NormSpace\_of\_BoundedLinearOperators(X,Y).$$

```
definition let X, Y be RealNormSpace;
func R_NormSpace_of_BoundedLinearOperators(X,Y) -> non empty NORMSTR equals
:: LOPBAN_1:def 15
  NORMSTR (# BoundedLinearOperators(X,Y),
        Zero_(BoundedLinearOperators(X,Y),
        R_VectorSpace_of_LinearOperators(X,Y)),
         Add_(BoundedLinearOperators(X,Y),
         R_VectorSpace_of_LinearOperators(X,Y)),
          Mult_(BoundedLinearOperators(X,Y),
          R_VectorSpace_of_LinearOperators(X,Y)),
          BoundedLinearOperatorsNorm(X,Y) #);
end;
```

If Y is a Banach space in this real normed space, we know that it is also *complete* and this is formalized as follows.

```
theorem :: LOPBAN_1:49
  for X be RealNormSpace
  for Y be RealBanachSpace holds
     R_NormSpace_of_BoundedLinearOperators(X,Y) is RealBanachSpace;

registration
  let X be RealNormSpace;
  let Y be RealBanachSpace;
  cluster R_NormSpace_of_BoundedLinearOperators (X,Y) -> complete;
end;
```

2.2 Banach Algebra Created by Linear Operators

If we restrict the formalization of the previous section to the case of X=Y and linear operators from X to X itself, syntheses consisting of f,g mappings of linear operators form multiplication operations on $BoundedLinearOperators(X,X)$ introduced earlier. Therefore, the identity mapping of X onto itself becomes the unit source for this multiplication. From this, we can introduce a normed ring structure which uses $BoundedLinearOperators(X,X)$ as a base. The formalization steps are shown below.

```
definition let X be RealNormSpace;
  func FuncMult(X) -> BinOp of BoundedLinearOperators(X,X) means
:: LOPBAN_2:def 4
  for f,g being Element of BoundedLinearOperators(X,X) holds
  it.(f,g) = f*g;
end;

theorem :: LOPBAN_2:3
  for X be RealNormSpace holds
    id (the carrier of X) is bounded LinearOperator of X,X;

definition let X be RealNormSpace;
   func FuncUnit(X) -> Element of BoundedLinearOperators(X,X) equals
:: LOPBAN_2:def 5
    id (the carrier of X);
  end;

theorem :: LOPBAN_2:4
   for X be RealNormSpace
   for f,g,h be bounded LinearOperator of X,X holds
   h = f*g iff for x be VECTOR of X holds h.x = f.(g.x);

theorem :: LOPBAN_2:5
  for X be RealNormSpace
  for f,g,h be bounded LinearOperator of X,X
  holds f*(g*h) =(f*g)*h;

theorem :: LOPBAN_2:6
  for X be RealNormSpace
  for f be bounded LinearOperator of X,X holds
  f*(id the carrier of X) = f  & (id the carrier of X )*f=f;

theorem :: LOPBAN_2:7
for X be RealNormSpace
for f,g,h be Element of BoundedLinearOperators(X,X)
holds f*(g*h) =(f*g)*h;
```

```
theorem :: LOPBAN_2:8
  for X be RealNormSpace
  for f be Element of BoundedLinearOperators(X,X) holds
  f*FuncUnit(X)= f & FuncUnit(X)*f=f;

theorem :: LOPBAN_2:9
  for X be RealNormSpace
  for f,g,h be Element of BoundedLinearOperators(X,X) holds
  f *(g+h)=f*g + f*h;

theorem :: LOPBAN_2:10
  for X be RealNormSpace
  for f,g,h be Element of BoundedLinearOperators(X,X) holds
  (g+h)*f = g*f + h*f;

theorem :: LOPBAN_2:11
for X be RealNormSpace
for f,g be Element of BoundedLinearOperators(X,X)
for a,b be Real holds
(a*b)*(f*g)=(a*f)*(b*g);

theorem :: LOPBAN_2:12
for X be RealNormSpace
for f,g be Element of BoundedLinearOperators(X,X)
for a be Real holds
a*(f*g) =(a*f)*g;

definition
let X be RealNormSpace;
  func Ring_of_BoundedLinearOperators(X) -> doubleLoopStr equals
:: LOPBAN_2:def 6
     doubleLoopStr
     (# BoundedLinearOperators(X,X),
           Add_(BoundedLinearOperators(X,X),
                    R_VectorSpace_of_LinearOperators(X,X)),
          FuncMult(X),
          FuncUnit(X),
          Zero_(BoundedLinearOperators(X,X),
                   R_VectorSpace_of_LinearOperators(X,X))
    #);
end;

definition
```

```
let X be RealNormSpace;
  func Ring_of_BoundedLinearOperators(X) -> doubleLoopStr equals
:: LOPBAN_2:def 6
     doubleLoopStr
     (# BoundedLinearOperators(X,X),
          Add_(BoundedLinearOperators(X,X),
                   R_VectorSpace_of_LinearOperators(X,X)),
         FuncMult(X),
         FuncUnit(X),
         Zero_(BoundedLinearOperators(X,X),
                  R_VectorSpace_of_LinearOperators(X,X))
    #);
end;
```

In continuation of the above formalization, we introduce $NormedAlgebra$ as a functor called $R\_Normed\_Algebra\_of\_BoundedLinearOperators(X)$ and particularly in the case of X being a Banach space, we formalize the fact that this space is a $BanachAlgebra$.

```
definition
let X be RealNormSpace;
func R_Normed_Algebra_of_BoundedLinearOperators(X) -> Normed_AlgebraStr
equals
:: LOPBAN_2:def 8

   Normed_AlgebraStr
    (# BoundedLinearOperators(X,X),
        FuncMult(X),
        Add_(BoundedLinearOperators(X,X),
                     R_VectorSpace_of_LinearOperators(X,X)),
        Mult_(BoundedLinearOperators(X,X),
                 R_VectorSpace_of_LinearOperators(X,X)),
        FuncUnit(X),
        Zero_(BoundedLinearOperators(X,X),
               R_VectorSpace_of_LinearOperators(X,X)),
       BoundedLinearOperatorsNorm(X,X)
          #);
end;
```

2.3 Functions of Linear Operators

As described at the onset, the primary objective of the authors for pursuing the work of formalizing the concepts of linear operators is to have the tools for a complete formalization of the Yoshida-Hille Theorem. The articles on functions of linear operators completed until now still cover only bounded linear operators.

For example, the following defines a geometric sequence of an element z in $Banach\_AlgebraX$.

```
definition
  let X be Banach_Algebra;
  let z be Element of X;
  func z GeoSeq -> sequence of X means
:: LOPBAN_3:def 13
  it.0 = 1.X & for n be Element of NAT  holds it.(n+1) = it.n * z;
end;

definition
  let X be Banach_Algebra;
  let z be Element of X, n be Element of NAT;
  func z #N n -> Element of X equals
:: LOPBAN_3:def 14
  z GeoSeq.n;
end;

theorem :: LOPBAN_3:44
   for X be Banach_Algebra
    for z be Element of X holds
  z #N 0 = 1.X;
```

Also, the following is a formalization of the Neumann Theorem which discusses the sufficient conditions for elements of $Banach\_AlgebraX$ to be reversible in multiplication.

```
theorem :: LOPBAN_3:45
for X be Banach_Algebra
for z be Element of X holds
||.z.|| < 1 implies z GeoSeq is summable norm_summable;

theorem :: LOPBAN_3:46
   for X be Banach_Algebra
  for x be Point of X
      st ||.1.X - x .|| < 1
  holds ( (1.X - x) GeoSeq is summable
  & (1.X - x) GeoSeq is norm_summable);

theorem :: LOPBAN_3:47
   for X be Banach_Algebra
  for x be Point of X
      st  ||.1.X - x .|| < 1
  holds x is invertible & x" = Sum ((1.X - x) GeoSeq );
```

We have also formalized the concept of exponential functions for elements of $Banach\_AlgebraX$ as shown below.

```
definition
let X be Banach_Algebra;
func exp_ X -> Function of the carrier of X, the carrier of X means
:: LOPBAN_4:def 10
  for z being Element of X holds it.z=Sum(z ExpSeq);
end;

definition let X,z;
  func exp z -> Element of X equals
:: LOPBAN_4:def 11
  (exp_ X).z;
end;

theorem :: LOPBAN_4:34
for z holds exp(z)=Sum(z ExpSeq);

theorem :: LOPBAN_4:35
for z1,z2 st z1,z2 are_commutative holds
     exp(z1+z2)=exp(z1) *exp(z2)
   &exp(z2+z1)=exp(z2) *exp(z1)
   &exp(z1+z2)=exp(z2+z1)
   &exp(z1),exp(z2) are_commutative;

theorem :: LOPBAN_4:36
  for z1,z2 st z1,z2 are_commutative holds
    z1* exp(z2)=exp(z2)*z1;

theorem :: LOPBAN_4:37
exp(0.X) = 1.X;

theorem :: LOPBAN_4:38
exp(z)*exp(-z)= 1.X & exp(-z)*exp(z)= 1.X;

theorem :: LOPBAN_4:39
  exp(z) is invertible & (exp(z))" = exp(-z)
&
exp(-z) is invertible & (exp(-z))" = exp(z);

theorem :: LOPBAN_4:40
for z for s,t be Real holds s*z,t*z are_commutative;

theorem :: LOPBAN_4:41
for z for s,t be Real holds
exp(s*z)*exp(t*z) = exp((s+t)*z) &
```

```
exp(t*z)*exp(s*z) = exp((t+s)*z) &
exp((s+t)*z) = exp((t+s)*z) &
exp(s*z),exp(t*z) are_commutative;
```

3. Conclusion

The primary objective of this work will be to completely formalize the materials of the Yoshida-Hille C_0 semi-group theorem. However, the theorems concerning linear operators alone compose a vast collection and will require a substantial amount of time to formalize. This work of collecting and formalizing the various facts and results concerning linear operators, which are applied in not only engineering and science, but in numerous fields, will be essential in developing the Mizar library and will form the backbone of our future work.

References

[1] Yasunari Shidama, *Banach Space of Bounded Linear Operators*, Formalized Mathematics, 12(**1**), 2004, pp.39-48.
[2] Yasunari Shidama, *The Banach Algebra of Bounded Linear Operators*, Formalized Mathematics, 12(**2**), 2004, pp.103-108.
[3] Yasunari Shidama, *The Series on Banach Algebra*, Formalized Mathematics, 12(**2**), 2004, pp.131-138.
[4] Yasunari Shidama, *The Exponential Function on Banach Algebra*, Formalized Mathematics, 12(**2**), 2004, pp.173-178.
[5] Noboru Endou, *Complex Banach Space of Bounded Linear Operators*, Formalized Mathematics, 12(**2**), 2004, pp.201-210.
[6] Noboru Endou, *Banach Algebra of Bounded Complex Linear Operators*, Formalized Mathematics, 12(**3**), 2004, pp.237-242.
[7] Noboru Endou, *Series on Complex Banach Algebra*, Formalized Mathematics, 12(**3**), 2004, pp.281-288.
[8] Noboru Endou, *Exponential Function on Complex Banach Algebra*, Formalized Mathematics, 12(**3**), 2004, pp.289-294.